

TUGAS AKHIR - KI141502

ESTIMASI BIAYA PERANGKAT LUNAK MENGUNAKAN METODE COCOMO YANG DIMODIFIKASI

Johannes Christian P. Sidabutar
NRP 5111100 179

Dosen Pembimbing
Prof. Drs. Ec. Ir. Riyanarto Sarno, M.Sc., Ph.D.
Sarwosri, S.Kom., M.T.

**JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2015**



FINAL PROJECT - KI141502

ESTIMATING SOFTWARE COST BASED ON MODIFIED COCOMO

Johannes Christian P. Sidabutar
NRP 5111100 179

Lecture Supervisors
Prof. Drs. Ec. Ir. Rianarto Sarno, M.Sc., Ph.D.
Sarwosri, S.Kom., M.T.

**DEPARTMENT OF INFORMATICS
Faculty of Information Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2015**

**ESTIMASI BLAYA PERANGKAT LUNAK
MENGUNAKAN METODE COCOMO
YANG DIMODIFIKASI
TUGAS AKHIR**

Dijadikan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada

Rampun Mata Kuliah Manajemen Informasi
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh
Johannes Christian P. Sidabutar
NRP : 5111 100 179

Disetujui oleh Dosen Pembimbing Tugas Akhir

Prof. Drs. Ec. Ir. Riyanarto Samudra, M.Sc., Ph.D.
NIP. 19590803 198601 1 001
(Pembimbing 1)

Sirwosri, S.Kom.M.T.
NIP. 19760809 200112 2 001
(Pembimbing 2)



**SURABAYA
JUNI, 2015**

ESTIMASI BIAYA PERANGKAT LUNAK MENGUNAKAN METODE COCOMO YANG DIMODIFIKASI

Nama Mahasiswa : Johannes Christian P. Sidabutar
NRP : 5111100179
Jurusan : Teknik Informatika FTIf-ITS
Dosen Pembimbing 1 : Prof. Drs. Ec. Ir. Rivanarto Sarno, M.Sc., Ph.D.
Dosen Pembimbing 2 : Sarwosri, S.Kom. M.T.

Abstrak

Dewasa ini, perangkat lunak merupakan sebuah kebutuhan bagi sebagian masyarakat. Tidak bisa dipungkiri, perangkat lunak dapat membantu manusia mempermudah segala pekerjaannya. Perangkat lunak diharapkan bisa memiliki kemampuan yang diharapkan oleh pemakainya. Seiring berkembangnya kebutuhan manusia, perangkat lunak juga diharapkan bisa berevolusi sesuai kebutuhan. Namun, evolusi perangkat tersebut harus bisa diandalkan dan memiliki performa yang baik. Dalam rangka mengatasi permasalahan tersebut diperlukan estimasi biaya yang mampu memperkirakan segala kebutuhan yang terdapat pada sebuah aplikasi.

Estimasi biaya perangkat lunak merupakan hal yang sangat penting dalam pengembangan aplikasi. Namun, banyak pengembang aplikasi kurang mempertimbangkan hal ini. Mereka hanya memperkirakan biaya aplikasi berdasarkan kebutuhan fungsional maupun metode atau algoritma yang digunakan. Model estimasi biaya aplikasi berbasis algoritma memiliki keterbatasan dalam mengatasi ketidakpastian dan kesalahan dalam tahap awal pengembangan aplikasi.

Tugas akhir ini berusaha untuk memecahkan permasalahan tersebut. Estimasi biaya yang dikembangkan meliputi model algoritma dan non algoritma. Metode algoritma dikembangkan dengan bantuan Constructive Cost Model

(COCOMO) yang merupakan teknik estimasi biaya perangkat lunak yang paling umum digunakan dalam pengembangan perangkat lunak. COCOMO bergantung pada beberapa variabel yang disebut dengan cost drivers. Cost drivers ini dikembangkan oleh beberapa ahli dan dikalkulasi menggunakan fungsi matematika. Namun dalam perkembangannya, untuk meningkatkan akurasi dari model ini diperlukan implementasi model non algoritma untuk mengatasi ketidakpastian dan kesalahan. Tugas akhir ini mengimplementasi fuzzy logic dan artificial neural network sebagai model non algoritma yang dapat memperbaiki akurasi COCOMO. Diharapkan kedua metode ini dapat membantu pengembang aplikasi dalam memperkirakan biaya aplikasi secara lebih akurat.

Kata Kunci : manajemen informasi, perangkat lunak, estimasi biaya perangkat lunak, COCOMO, fuzzy logic, artificial neural network, cost drivers.

“ESTIMATING SOFTWARE COST BASED ON MODIFIED COCOMO.”

Student Name : Johannes Christian P
Student ID : 5111100179
Major : Teknik Informatika FTIf-ITS
Advisor 1 : Prof. Drs. Ec. Ir. Riyanarto Sarno, M.Sc., Ph.D.
Advisor 2 : Sarwosri, S.Kom., M.T.

Abstract

Recently, software has become needs for some people. It cannot be denied, software could help people to ease their activities. Software is expected to have ability as what the user hoped for. As the developing of people needs, software also is anticipated to evolve as the requirement. But, the software evolution must be reliable and have good performance. In order to handle the problem, it is needed cost estimation that could manage every requirement which software has.

Software cost estimation is one of critical tasks in software development. But, some developers less consider this issue. They estimate software cost based one algorithmic models. These models are limited by their inability to cope with uncertainties and imprecision in early stage of software development.

This final project tries to handle the problems. Cost estimation which is implemented based on algorithmic and non alorithmic models. Algorithmic model is applied with help of Constructive Cost Model (COCOMO) which is most common software cost estimation model. COCOMO depends on several variables or cost drivers. Cost drivers is designed by some experts an calculated with mathematical functions. But, in order to improve the accuracy, the model requires implementation of non algorithmic models which could help to overcome uncertainties and imprecision. This final project implements fuzzy logic and artificial neural network as non algorithmic models which could

improve the accuracy of COCOMO. It is expected that these models could help software developer to estimate software cost accurately.

Keywords: information management, software, software cost estimation, COCOMO, fuzzy logic, artificial neural network, cost drivers.

KATA PENGANTAR

Puji syukur kepada Tuhan Yang Maha Esa atas segala karunia dan rahmat-Nya penulis dapat menyelesaikan tugas akhir yang berjudul:

“Estimasi Biaya Perangkat Lunak Menggunakan Metode COCOMO yang Dimodifikasi”

Harapan dari penulis semoga apa yang tertulis di dalam buku tugas akhir ini dapat bermanfaat bagi pengembangan ilmu pengetahuan saat ini, serta dapat memberikan kontribusi yang nyata.

Dalam pelaksanaan dan pembuatan tugas akhir ini tentunya sangat banyak bantuan yang penulis terima dari berbagai pihak, tanpa mengurangi rasa hormat penulis ingin mengucapkan terima kasih sebesar-besarnya kepada:

1. Keluarga penulis yang selalu memberikan dukungan penuh dan memberikan semangat untuk menyelesaikan tugas akhir ini.
2. Bapak Prof. Riyanarto Sarno selaku pembimbing pertama yang membantu penulis memecahkan setiap permasalahan yang ada dan memberikan petunjuk selama pengerjaan tugas akhir ini. Penulis juga berterima kasih untuk kesediaan pembimbing dalam meluangkan waktu dalam proses menyelesaikan tugas akhir ini.
3. Ibu Sarwosri selaku pembimbing kedua yang telah bersedia meluangkan waktu untuk memberikan petunjuk selama proses pengerjaan tugas akhir ini.
4. Bapak Victor Hariadi selaku dosen wali penulis, serta segenap dosen Jurusan Teknik Informatika ITS yang telah banyak memberikan ilmu dan bimbingan yang tak ternilai harganya bagi penulis.
5. Seluruh staf dan karyawan FTIf ITS yang banyak memberikan kelancaran administrasi akademik kepada penulis.

6. Nadia Emeraldalda Christiana yang selalu memberi dorongan dan semangat.
7. Teman-teman penghuni Lab IGS dan MI yang telah memberikan banyak dukungan dan semangat serta tempat kepada penulis.
8. Teman-teman angkatan 2011 jurusan Teknik Informatika ITS yang telah menemani perjuangan selama 4 tahun ini atas saran, masukan, dan dukungan terhadap pengerjaan tugas akhir ini.
9. Serta pihak-pihak lain yang tidak dapat disebutkan disini yang telah banyak membantu penulis dalam penyusunan tugas akhir ini.

Penulis telah berusaha sebaik-baiknya dalam menyusun tugas akhir ini, namun penulis mohon maaf apabila terdapat kekurangan, kesalahan maupun kelalaian yang telah penulis lakukan. Kritik dan saran yang membangun dapat disampaikan sebagai bahan perbaikan selanjutnya.

Surabaya, Juni 2015

Johannes Christian P

DAFTAR ISI

LEMBAR PENGESAHAN	v
Abstrak	vii
<i>Abstract</i>	ix
KATA PENGANTAR.....	xi
DAFTAR ISI.....	xiii
DAFTAR GAMBAR	xvii
DAFTAR TABEL	xix
BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Tujuan	2
1.3 Rumusan Permasalahan.....	2
1.4 Batasan Permasalahan	2
1.5 Metodologi.....	2
1.6 Sistematika Penulisan.....	4
BAB II TINJAUAN PUSTAKA	7
2.1. Penelitian Terkait	7
2.2. Perangkat Lunak.....	7
2.3. <i>Software Cost Estimation</i>	8
2.3.1. <i>Algorithmic models</i>	9
2.3.2. <i>Non-algorithmic models</i>	9
2.4. COCOMO	10
2.4.1. COCOMO 81	10
2.4.2. COCOMO II.....	14

2.5. <i>Fuzzy Logic</i>	24
2.5.1. <i>Fuzzification Process</i>	25
2.5.2. <i>Inference Process</i>	26
2.5.3. <i>Defuzzification Process</i>	27
2.6. <i>Neural Network</i>	28
BAB III ANALISIS DAN PERANCANGAN SISTEM	31
3.1. Analisis	31
3.1.1. Analisis Permasalahan	31
3.1.2. Data Uji	33
3.1.3. Optimasi COCOMO dengan <i>Fuzzy Logic</i>	36
3.1.4. Pendekatan Metode <i>Fuzzy</i> pada COCOMO	37
3.1.5. Optimasi COCOMO dengan <i>Neural Network</i>	40
3.1.6. Pendekatan <i>Fuzzy</i> dan <i>Artificial Neural Network</i> pada COCOMO	47
3.1.7. Deskripsi Umum Aplikasi berbasis Metode yang Dikembangkan	47
3.2. Perancangan Sistem	53
3.2.1. Perancangan Antarmuka Pengguna	53
BAB IV IMPLEMENTASI	55
4.1. Lingkungan Implementasi	55
4.2. Penjelasan Implementasi	55
4.3. Implementasi <i>Fuzzy</i> pada COCOMO II	56
4.4. Implementasi <i>Neural Network</i> pada <i>Fuzzy</i> COCOMO II ...	65
4.5. Implementasi Aplikasi Estimasi Biaya Perangkat Lunak ...	67
BAB V PENGUJIAN DAN EVALUASI	69
5.1. Lingkungan Uji Coba	69

5.2. Metode Pengujian.....	69
5.3. Evaluasi COCOMO 81, COCOMO II, dan Fuzzy COCOMO II.....	70
5.4. Evaluasi <i>Neural Network</i> pada <i>Fuzzy COCOMO</i>	71
5.5. Uji Coba pada Aplikasi Estimasi Biaya Perangkat Lunak ..	74
5.5.1. Uji Coba Memasukkan Nilai Atribut	74
5.5.2. Uji Coba Melihat Nilai Estimasi Biaya Perangkat Lunak ..	76
BAB VI KESIMPULAN DAN SARAN	79
6.1 Kesimpulan	79
6.2 Saran	79
DAFTAR PUSTAKA.....	81
LAMPIRAN.....	85
A. COCOMO Dataset	85
B. NASA Dataset.....	89
D. Hasil estimasi biaya perangkat lunak menggunakan metode COCOMO 81	111
E. Hasil estimasi biaya perangkat lunak menggunakan metode COCOMO II.	117
F. Hasil estimasi biaya perangkat lunak menggunakan metode <i>Fuzzy-COCOMO II</i>	123
G. Daftar Istilah	128
BIODATA PENULIS.....	130

(Halaman ini sengaja dikosongkan)

DAFTAR TABEL

Tabel 2. 1. Konstanta pada setiap mode di COCOMO 81	11
Tabel 2. 2. Kategori <i>Effort Multiplier</i>	12
Tabel 2. 3. Nilai dari COCOMO <i>effort multiplier</i>	13
Tabel 2. 4. Perbedaan COCOMO 81 dan COCOMO II	15
Tabel 3. 1 Atribut COCOMO Dataset	34
Tabel 3. 2. Atribut NASA 93 Dataset	35
Tabel 3. 3. Daftar Kebutuhan Fungsional Perangkat Lunak	48
Tabel 3. 4. Daftar Kode Diagram Kasus Penggunaan	49
Tabel 3. 5. Spesifikasi Kasus Pengguna Memasukkan Nilai Atribut	50
Tabel 3. 6. Spesifikasi Kasus Pengguna Melihat Estimasi	52
Tabel 4. 1. COCOMO II <i>Effort Multiplier</i>	63
Tabel 4. 2. <i>Fuzzy</i> COCOMO II <i>Effort Multiplier</i>	64
Tabel 5. 1. Perbandingan MMRE terhadap COCOMO 81, COCOMO II dan <i>fuzzy</i> COCOMO II	70
Tabel 5. 2. Perbandingan MRE COCOMO dengan <i>neural network</i> pada <i>dataset</i> COCOMO	72
Tabel 5. 3. Perbandingan MRE COCOMO dengan <i>neural network</i> pada <i>dataset</i> COCOMO	73
Tabel 5. 4. <i>Blackbox Testing</i> Kasus Pengguna Memasukkan Nilai Atribut	74
Tabel 5. 5. <i>Blackbox</i> Testing Kasus Pengguna Memasukkan Nilai Atribut	77

DAFTAR GAMBAR

Gambar 2. 1. Deskripsi RELY.....	17
Gambar 2. 2. Deskripsi DATA.....	17
Gambar 2. 3. Deskripsi RUSE.....	17
Gambar 2. 4. Deskripsi DOCU.....	18
Gambar 2. 5. Deskripsi TIME	18
Gambar 2. 6. Deskripsi STOR.....	19
Gambar 2. 7. Deskripsi PVOL.....	19
Gambar 2. 8. Deskripsi ACAP	19
Gambar 2. 9. Deskripsi PCAP.....	20
Gambar 2. 10. Deskripsi PCON	20
Gambar 2. 11. Deskripsi APEX.....	20
Gambar 2. 12. Deskripsi PLEX	21
Gambar 2. 13. Deskripsi LTEX.....	21
Gambar 2. 14. Deskripsi TOOL	22
Gambar 2. 15. Deskripsi SITE.....	22
Gambar 2. 16. Deskripsi SCED.....	22
Gambar 2. 17. Deskripsi CPLX.....	23
Gambar 2. 18. Nilai dari <i>scale factors</i>	24
Gambar 2. 19. <i>Fuzzy Inference System</i>	24
Gambar 2. 20. <i>Triangular Membership Function</i>	25
Gambar 2. 21. <i>Trapezoidal Membership Function</i>	26
Gambar 2. 22. <i>Gaussian Membership Function</i>	26
Gambar 2. 23. <i>Defuzzification Process</i>	28
Gambar 2. 24. <i>Single-layer two-input perceptron</i>	29
Gambar 3. 1. Arsitektur Fuzzy Logic untuk COCOMO	37
Gambar 3. 2. Representasi DATA menggunakan <i>Gaussian Membership Function</i>	39
Gambar 3. 3. <i>Multilayer Neural Network</i>	41
Gambar 3. 4 . Diagram Alur Implementasi <i>Basic Architecture</i> ..	42
Gambar 3. 5. Arsitektur <i>Basic Neural Network</i>	43
Gambar 3. 6. Diagram Alur Implementasi <i>Modified Architecture</i>	45
Gambar 3. 7. Arsitektur <i>Modified Neural Network</i>	46

Gambar 3. 8. Diagram Alur Proses Implementasi Fuzy Logic dan Neural Network pada COCOMO II	47
Gambar 3. 9. Diagram Kasus Penggunaan Sistem.....	49
Gambar 3. 10. Diagram Aktifitas Menghitung Biaya Perangkat Lunak.....	51
Gambar 3. 11. Diagram Aktifitas Menampilkan estimasi biaya perangkat lunak.....	52
Gambar 3. 12. Antarmuka Aplikasi Estimasi Biaya Perangkat Lunak.....	53
Gambar 4. 1. FIS <i>Editor</i>	57
Gambar 4. 2. <i>Membership Function</i> pada FIS <i>Editor</i>	57
Gambar 4. 3. <i>Membership Function Editor</i>	58
Gambar 4. 4. Deskripsi LTEX.....	59
Gambar 4. 5. <i>LTEX Membership Function</i>	59
Gambar 4. 6. <i>Output</i> pada <i>Membership Function Editor</i>	60
Gambar 4. 7. Nilai LTEX <i>Effort Multiplier</i>	61
Gambar 4. 8. LTEX <i>Membership Function</i>	61
Gambar 4. 9. <i>Rule Editor</i> LTEX.....	62
Gambar 4. 10. <i>Rule Viewer</i> LTEX	63
Gambar 5. 1. Grafik perbandingan MMRE terhadap COCOMO 81, COCOMO II dan <i>fuzzy</i> COCOMO II.....	71
Gambar 5. 2. Peringatan field masih ada yang belum diisi pada aplikasi.....	76
Gambar 5. 3. Aplikasi menampilkan jumlah pegawai, lama serta biaya perangkat lunak	77

BAB I

PENDAHULUAN

Pada bab ini akan dipaparkan mengenai garis besar tugas akhir yang meliputi latar belakang, tujuan, rumusan dan batasan permasalahan, metodologi pembuatan tugas akhir, dan sistematika penulisan.

1.1 Latar Belakang

Perangkat lunak merupakan sebuah kebutuhan bagi sebagian masyarakat. Tidak bisa dipungkiri, perangkat lunak dapat membantu manusia mempermudah segala pekerjaannya. Perangkat lunak diharapkan bisa memiliki kemampuan yang diharapkan oleh pemakainya. Seiring berkembangnya kebutuhan manusia, perangkat lunak juga diharapkan bisa berevolusi sesuai kebutuhan. Namun, evolusi perangkat tersebut harus bisa diandalkan dan memiliki performa yang baik. Dalam rangka mengatasi permasalahan tersebut diperlukan estimasi biaya yang mampu memperkirakan segala kebutuhan yang terdapat pada sebuah aplikasi.

Saat ini, teknik estimasi yang paling umum digunakan adalah *Constructive Cost Model* (COCOMO). COCOMO merupakan teknik estimasi biaya aplikasi berbasis pada metode algoritma. Namun dalam perkembangannya, untuk meningkatkan akurasi dari model ini diperlukan implementasi model non algoritma untuk mengatasi ketidaktentuan dan kesalahan dalam tahap awal pengembangan aplikasi.

Untuk mengatasi segala permasalahan diatas, tugas akhir ini menggabungkan metode algoritma dan non algoritma dalam memprediksi biaya perangkat lunak. Hal ini dilakukan dengan cara mengimplementasi metode *fuzzy logic* dan *artificial neural network* pada COCOMO. Dengan penggabungan metode ini diharapkan biaya aplikasi dapat diestimasi dengan akurat dan dapat mengatasi permasalahan ketidakpastian dan kesalahan.

1.2 Tujuan

Tujuan dari pembuatan tugas akhir ini adalah membangun model estimasi biaya perangkat lunak secara lebih akurat yang dapat mengatasi ketidakpastian dan kesalahan yang mungkin terjadi.

1.3 Rumusan Permasalahan

Berikut adalah beberapa rumusan masalah yang terdapat dalam rancang bangun sistem ini:

1. Bagaimana menemukan model yang tepat dalam estimasi biaya aplikasi?
2. Bagaimana memodelkan kebutuhan aplikasi pada COCOMO?
3. Bagaimana meningkatkan akurasi pada COCOMO dengan mengimplementasi metode non algoritma?

1.4 Batasan Permasalahan

Permasalahan yang dibahas dalam tugas akhir ini memiliki beberapa batasan, di antaranya sebagai berikut:

1. Mengimplementasikan *fuzzy logic* dan *artificial neural network* ke dalam COCOMO dalam rangka meningkatkan akurasi estimasi biaya perangkat lunak.
2. Data uji yang digunakan berdasarkan data uji yang telah ada yang berasal dari COCOMO dan NASA.

1.5 Metodologi

Langkah-langkah yang ditempuh dalam pengerjaan tugas akhir ini yaitu:

1. Studi literatur

Pada tahap ini dilakukan pengumpulan informasi

mengenai bagaimana memodelkan biaya aplikasi, model apa saja yang berkembang, serta bagaimana meningkatkan akurasi estimasi biaya aplikasi.

2. Analisis

Pada tahap ini dilakukan analisis yang bertujuan untuk mengetahui sifat atau kecenderungan dari COCOMO serta mencari metode untuk menambahkan metode lain dalam rangka memperbaiki akurasi.

3. Implementasi

Pada tahap ini dilakukan pembuatan elemen estimasi biaya perangkat lunak serta perbaikan estimasi melalui metode non algoritma

Perincian tahap ini adalah sebagai berikut:

- a. Implementasi estimasi biaya aplikasi menggunakan COCOMO
- b. Implementasi *fuzzy logic* pada COCOMO.
- c. Implementasi *artificial neural network* pada COCOMO.

4. Pengujian dan evaluasi

Pada tahap ini dilakukan pengujian terhadap model yang dikembangkan dengan memakai data uji yang telah dipersiapkan.

5. Penyusunan buku tugas akhir

Pada tahap ini dilakukan pendokumentasian dan pelaporan dari seluruh konsep, dasar teori, implementasi, proses yang telah dilakukan, dan hasil-hasil yang telah didapatkan selama pengerjaan tugas akhir.

1.6 Sistematika Penulisan

Buku tugas akhir ini bertujuan untuk mendapatkan gambaran dari pengerjaan tugas akhir ini. Selain itu, diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut. Secara garis besar, buku tugas akhir terdiri atas beberapa bagian seperti berikut ini.

BAB I. PENDAHULUAN

Bab ini berisi mengenai latar belakang, tujuan, dan manfaat dari pembuatan tugas akhir. Selain itu, permasalahan, batasan masalah, metodologi yang digunakan, dan sistematika penulisan juga merupakan bagian dari bab ini.

BAB II. TINJAUAN PUSTAKA

Bab ini berisi penjelasan secara detail mengenai dasar-dasar penunjang dan teori-teori yang digunakan untuk mendukung pembuatan tugas akhir ini.

BAB III. METODE PEMECAHAN MASALAH

Pada bab ini akan dibahas mengenai metodologi pemecahan masalah yang digunakan sebagai dasar solusi dari pembuatan Tugas Akhir.

BAB IV. ANALISIS DAN PERANCANGAN SISTEM

Bab ini membahas tahap analisis permasalahan dan perancangan dari sistem yang akan dibangun. Analisis permasalahan membahas permasalahan yang diangkat dalam pengerjaan tugas akhir.

BAB IV. IMPLEMENTASI

Bab ini membahas implementasi dari desain yang telah dibuat pada bab sebelumnya. Bab ini berisi proses implementasi dari setiap kelas pada semua modul.

BAB V. PENGUJIAN DAN EVALUASI

Bab ini menjelaskan kemampuan perangkat lunak dengan melakukan pengujian kebenaran dan pengujian kinerja dari sistem yang telah dibuat.

BAB VI. KESIMPULAN DAN SARAN

Bab ini merupakan bab terakhir yang menyampaikan kesimpulan dari hasil uji coba yang dilakukan dan saran untuk pengembangan perangkat lunak ke depannya.

BAB II

TINJAUAN PUSTAKA

Pada bab ini akan dibahas mengenai teori-teori yang menjadi dasar dari pembuatan tugas akhir.

2.1. Penelitian Terkait

Reddy dan Raju [1] melakukan sebuah penelitian untuk meningkatkan akurasi dengan cara melakukan proses *fuzzifying* pada COCOMO *cost drivers*. Mereka mempelajari sifat dari COCOMO *cost drivers*, kemudian melakukan pendekatan menggunakan karakter dari *Gaussian Membership Function* pada *cost drivers* dalam rangka merepresentasi nilai linguistik dari setiap variabel. Hasil penelitian yang dilakukan menunjukkan bahwa model yang dikembangkan lebih akurat dibanding model asli dari COCOMO maupun *Trapezoidal Membership Function (TMF)*.

Kaushik dan Soni [2] mengobservasi penggunaan dari *back-propagation neural networks* dalam estimasi biaya perangkat lunak. Mereka meningkatkan akurasi dari model COCOMO dengan cara melakukan proses training terhadap arsitektur *neural network* untuk setiap *dataset*. Hasil penelitian menunjukkan pendekatan dengan *neural network* memiliki akurasi yang lebih baik dibanding COCOMO.

2.2. Perangkat Lunak

Perangkat lunak adalah instruksi (program komputer) yang bila dieksekusi dapat menjalankan fungsi tertentu, struktur data yang dapat membuat program memanipulasi informasi, dokumen yang menjelaskan program [2]. Teknik implementasi perangkat lunak disebut dengan *software engineering*. *Software Engineering* adalah pendekatan aplikasi yang sistematis, disiplin, terukur untuk mengembangkan, melakukan operasi dan perbaikan perangkat lunak [3]. Adapun hal-hal yang penting dalam rekayasa perangkat lunak adalah :

- a. Ketepatan waktu (*on time*)
- b. Kesesuaian biaya (*on budget*)
- c. Kemampuan yang memadai (*acceptable performance*)
- d. Operasi yang sesuai (*correct operation*)

Dalam *software engineering* tidak hanya diperlukan ilmu komputer yang mumpuni namun juga diperlukan ilmu manajemen yang handal. Ilmu manajemen proyek meliputi semua hal yang berkaitan dengan proyek, seperti ruang lingkup proyek, anggaran, tenaga kerja, kualitas, manajemen resiko dan keandalan, perbaikan kualitas, dan metode-metode kuantitatif. Secara spesifik, ilmu manajemen ini berperan dalam menciptakan ketepatan waktu dan kesesuaian biaya. Dalam pengembangannya dikenal dengan 2 model umum dalam pengembangan aplikasi yaitu *Waterfall Model* dan *Spiral Model*.

2.3. *Software Cost Estimation*

Software cost atau biaya perangkat lunak adalah anggaran dana yang dikeluarkan dalam pengembangan sebuah aplikasi. *Software cost estimation* memiliki arti memprediksi total tenaga (*effort*), waktu dan tingkat kemampuan karyawan yang dibutuhkan dalam pengembangan sebuah perangkat lunak [4]. Estimasi ini sangat penting dalam perencanaan sebuah perangkat lunak dikarenakan permasalahan utama umumnya terjadi dalam tiga bulan pengembangan perangkat lunak menjadi dua yaitu *algorithmic* dan *non-algorithmic model*. yang dikarenakan jadwal yang terburu-buru, kurangnya komitmen, dan tidak profesional dalam teknik estimasi biaya [5].

Estimasi biaya yang terbaik adalah perkiraan yang dibuat di tahap awal pengembangan aplikasi, yang didasarkan pada studi kelayakan dan dokumen spesifikasi kebutuhan [6]. Akan tetapi, estimasi pada tahap awal pengembangan aplikasi sangat sulit dilakukan, umumnya estimasi ini kurang akurat dikarenakan hanya sedikit yang dapat diketahui atau diprediksi pada tahap awal.

Estimasi biaya serta jadwal dalam pengembangan perangkat lunak berguna untuk perencanaan dan pemantauan

sebuah aplikasi. Teknik estimasi perangkat lunak dapat dikategorikan menjadi dua yaitu *algorithmic* dan *non-algorithmic model*.

2.3.1. *Algorithmic models*

Barry W. Boehm adalah peneliti pertama yang melihat *software engineering* dari sudut pandang ekonomi kemudian melahirkan model estimasi biaya perangkat lunak. Pada tahun 1981, COCOMO-81 diciptakan setelah mempelajari basis data yang besar dari perusahaan TRW Inc. yaitu perusahaan besar yang bergerak dibidang otomotif dan kedirgantaraan [7]. Putnam juga mengembangkan teknik estimasi biaya perangkat lunak yang disebut dengan SLIM pada tahun 1978 [8]. Baik COCOMO maupun SLIM, keduanya memakai jumlah kode program (*Line of Code*) sebagai masukan yang penting pada teknik estimasi yang mereka kembangkan.

Algorithmic model ialah teknik estimasi model yang bergantung pada data biaya pengembangan aplikasi terdahulu. Namun, model ini memiliki keterbatasan dikarenakan atribut dan *relationships* yang digunakan untuk memprediksi biaya perangkat lunak bisa berubah setiap saat, hal ini dapat menciptakan perbedaan pada ruang lingkup pengembangan perangkat lunak [9].

Dalam perkembangannya, model ini dinilai gagal untuk menciptakan solusi yang tepat sesuai dengan perkembangan teknologi. Model ini hanya dapat berhasil pada sebagian kecil model pengembangan perangkat lunak tapi tidak bisa diterapkan pada model pengembangan yang lain.

2.3.2. *Non-algorithmic models*

Pada tahun 1990 peneliti merubah perhatian dalam estimasi biaya perangkat lunak menjadi berbasis *soft computing*. *Soft computing* adalah segolongan metoda yang mampu mengolah data dengan baik walaupun didalamnya terdapat ketidakpastian, ketidakakuratan maupun kebenaran parsial [10]. *Soft computing*

dapat dikategorikan ke dalam tiga tipe. Mereka adalah *fuzzy logic*, *neural networks* dan *probabilistic reasoning*. Hal yang penting yang terdapat dalam *soft computing* adalah metode ini bersifat komplementer dan sinergis, bukan kompetitif. Artinya, metode ini bersifat saling melengkapi dan bekerja sama dalam mengatasi permasalahan yang bersifat tidak pasti atau ambigu. Tujuan penggunaan metode ini adalah untuk mencari toleransi pada ketidaktepatan, ketidakpastiaan, perkiraan dan kebenaran yang parsial untuk mencapai solusi yang diinginkan.

2.4. COCOMO

Cost Constructive Model (COCOMO) adalah teknik estimasi perangkat lunak menggunakan *source of line code* (SLOC) yang digunakan untuk mengestimasi biaya, tenaga yang diperlukan, dan penjadwalan dalam pengembangan perangkat lunak. COCOMO merupakan model estimasi perangkat lunak yang menggunakan *algorithmic model* dimana model ini dikembangkan dengan cara mengumpulkan banyak data dari perangkat lunak yang sudah pernah dikerjakan. Data ini dianalisa untuk menemukan formula yang tepat sesuai dengan hasil observasi. Dalam perkembangannya COCOMO terbagi 2 jenis yaitu COCOMO 81 dan COCOMO II. Hasil dari estimasi biaya disebut *person-month* yang berarti jumlah usaha yang harus dilakukan pegawai dalam suatu periode bulan. Pada COCOMO dikenal istilah *cost drivers* yang memiliki arti variabel atau parameter yang digunakan dalam estimasi biaya perangkat lunak.

2.4.1. COCOMO 81

COCOMO 81 pertama kali diterbitkan pada tahun 1981 oleh Barry Boehm W. [11] sebagai model untuk memperkirakan usaha, biaya, dan jadwal untuk proyek-proyek perangkat lunak. Ini menarik pada studi dari 63 proyek di TRW Aerospace di mana Barry Boehm adalah Direktur Riset dan Teknologi Perangkat Lunak pada tahun 1981. Penelitian ini memeriksa proyek-proyek

yang memiliki ukuran mulai dari 2.000 sampai 100.000 baris kode dan bahasa pemrograman yang digunakan. Proyek-proyek ini didasarkan pada model pengembangan perangkat lunak dengan *waterfall model* yang merupakan proses pengembangan perangkat lunak yang umum pada tahun 1981. Model ini memiliki tiga mode yang dibagi berdasarkan lingkup kerja, ukuran dari perangkat lunak, dan batasan-batasan yang terkait proyek perangkat lunak. Ketiga mode tersebut yaitu:

- a. *Organic*: mode ini digunakan untuk perangkat lunak dengan sedikit pengembang atau karyawan yang terlibat, perangkat lunak juga umum diimplementasikan, dan biasanya pengembangannya bersifat rumahan atau skala kecil.
- b. *Embedded*: mode ini digunakan untuk perangkat lunak yang memiliki batasan yang sangat jelas di mana perangkat lunak ini sangat terikat dengan perangkat keras, perangkat lunak, regulasi dan prosedur operasional yang kompleks.
- c. *Semi-detached*: mode ini berada diantara mode *organic* dan *embedded*. Biasanya perangkat lunak pada mode ini memiliki baris kode mencapai 300000 baris.

Mode yang terdapat diatas dipergunakan sebagai acuan dalam penentuan konstanta. Tabel 2.1. memperlihatkan daftar konstanta berdasarkan mode yang digunakan.

Tabel 2. 1. Konstanta pada setiap mode di COCOMO 81

Model	A	B	c	D
Organic	2.4	1.05	2.5	0.38
Semi-detached	3.0	1.12	2.5	0.35
Embeded	3.6	1.2	2.5	0.32

Persamaan 2.1. menampilkan persamaan yang digunakan untuk menghitung usaha serta waktu yang dibutuhkan pada COCOMO 81 .

$$PM = a * (KDSI)^b * \prod_i^p EM_i$$

$$TDEV = c * (PM)^d \quad \dots(2.1.)$$

Dimana:

- PM adalah usaha dalam *person-month*.
- EM adalah perkalian dari setiap variabel *effort multiplier*.
- TDEV adalah waktu yang diperlukan untuk mengembangkan sebuah perangkat lunak.
- KDSI adalah jumlah baris kode (dalam ribuan).
- a,b,c,d merupakan konstanta yang didapat tergantung pada mode yang digunakan.

Pada COCOMO 81 terdapat 15 *effort multiplier* yang terbagi dalam empat kategori. *Effort multiplier* ini merupakan *linguistic variable* (variabel verbal) yang mendeskripsikan parameter yang digunakan dalam pengembangan perangkat lunak. Setiap *effort multiplier* memiliki skala dari Very Low sampai Extra High dimana setiap skala ini memiliki nilai tersendiri yang nantinya dipergunakan untuk menghitung usaha yang diperlukan. Tabel 2.2. menampilkan daftar *effort multiplier* beserta kategorinya. Sedangkan Tabel 2.3 menampilkan nilai setiap skala pada *effort multiplier*.

Tabel 2. 2. Kategori *Effort Multiplier*

Kategori	Effort Multiplier
Product Attributes	RELY - Required reliability
	DATA - Database size
	CPLX - Product complexity
Computer Attributes	TIME – Execution time constraint
	STOR – Main storage constraint
	VIRT – Virtual machine volatility

Kategori	Effort Multiplier
	TURN – Computer turnaround time
Personnel Attributes	ACAP – Analyst capability
	PCAP – Programmer capability
	VEXP - Virtual machine experience
	AEXP – Application experience
	LEXP - Programming language experience
Project Attributes	MODP - Modern programming practices
	TOOL - Use of software tools
	SCED - Required development schedule

Tabel 2. 3. Nilai dari COCOMO *effort multiplier*

Effort Multiplier	Very Low	Low	Nominal	High	Very High	Extra High
ACAP	1.46	1.19	1.00	0.86	0.71	-
PCAP	1.42	1.17	1.00	0.86	0.70	-
AEXP	1.29	1.13	1.00	0.91	0.82	-
MODP	1.24	1.10	1.00	0.91	0.82	-
TOOL	1.24	1.10	1.00	0.91	0.83	-
VEXP	1.21	1.10	1.00	0.90	-	-
LEXP	1.14	1.07	1.00	0.95	-	-
SCED	1.23	1.08	1.00	1.04	1.10	-
STOR	-	-	1.00	1.06	1.21	1.56
DATA	-	0.94	1.00	1.08	1.16	-
TIME	-	-	1.00	1.11	1.30	1.66
TURN	-	0.87	1.00	1.07	1.15	-
VIRT	-	0.87	1.00	1.15	1.30	-
RELY	0.75	0.88	1.00	1.15	1.40	-
CPLX	0.70	0.85	1.00	1.15	1.30	1.65

Penjelasan dari masing masing effort multiplier akan dipaparkan pada bab 2.4.2 dikarenakan *effort multiplier* pada COCOMO 81 dan COCOMO II memiliki kemiripan.

2.4.2. COCOMO II

Pada July 1994, B. Boehm [12] memulai pengembangan proyek COCOMO II dengan tujuan untuk mengakomodir pendekatan yang lebih baik terhadap estimasi biaya perangkat lunak. Berbeda dengan COCOMO 81, COCOMO II menggunakan *spiral model* sebagai siklus (*life-cycle*) dalam pengembangan perangkat lunak. Dengan model ini dimungkinkan untuk menghitung biaya evolusi perangkat lunak. Artinya apabila perangkat lunak sudah selesai dikerjakan dan pada perkembangannya dikehendaki adanya evolusi atau perubahan, maka estimasi biaya evolusi perangkat lunak dapat dilakukan. Hal ini dikarenakan dalam pengembangan perangkat lunak sangat penting untuk dapat mengakomodir evolusi dengan cepat [13]. Model ini dikembangkan dari 161 data proyek perangkat lunak. Tabel 2.4. menampilkan perbedaan COCOMO 81 dengan COCOMO II.

Tabel 2. 4. Perbedaan COCOMO 81 dan COCOMO II

Kategori	COCOMO 81	COCOMO II
Eksponen	Eksponen = konstanta yang mutlak yang dipilih berdasar <i>mode</i> - Organic = 1.05 - Semi-detached = 1.12 - Embedded = 1.20	Eksponen = nilai diambil dari lima <i>scale factor</i> - PREC - FLEX - RESL - TEAM - PMAT
<i>Dataset</i>	Terdiri dari 63 <i>data point</i>	Terdiri dari 161 <i>data point</i>
Baris Kode	KDSI	KLOC
<i>Effort Multiplier</i>	Terdapat 15 <i>Effort Multiplier</i>	Terdapat 17 <i>Effort Multiplier</i>

Sama halnya dengan pendahulunya, model ini menggunakan beberapa *cost drivers* yaitu jumlah baris kode (KSLOC), 17 *Effort Multiplier*, dan 5 *Scale Factors*. COCOMO II bagi kedalam 3 mode, yaitu:

- The Applications Composition Model*: digunakan pada tahap awal pengembangan perangkat lunak dan estimasi usaha pada perangkat lunak menggunakan besaran yang diestimasi berdasar pada estimasi sederhana dan formula produktifitas.
- The Early Design Model*: digunakan setelah kebutuhan perangkat lunak dan arsitektur perangkat lunak telah stabil.
- The Post Architecture Model*: digunakan pada saat implementasi perangkat lunak dan tahap estimasi usaha. Mode ini menggunakan parameter secara menyeluruh yang merepresentasikan kemampuan karyawan, perangkat lunak serta karakter dari proyek tersebut.

The Post Architecture Model merupakan model yang paling lengkap dari COCOMO II. Persamaan 2.2. menjelaskan bagaimana cara mendapatkan usaha dalam *person month* sedangkan Persamaan 2.3. menjelaskan rumus yang digunakan dalam mendapatkan lama pengerjaan dalam bulan.

$$PM = A * (KSLOC)^{B+C} * \prod_{i=1}^{17} EM_i$$

$$C = 0.01 * \sum_{i=1}^5 SF_i$$

...(2.2.)

$$TDEV = C * (PM)^F$$

$$F = D + 0.2 * 0.01 * \sum_{i=1}^5 SF_i = D + 0.2 * (E - B)$$

...(2.3.)

Dimana:

- a. PM adalah usaha dalam *person-month*.
- b. TDEV adalah waktu yang diperlukan untuk mengembangkan sebuah perangkat lunak.
- c. EM adalah perkalian dari setiap variabel *effort multiplier*.
- d. SF adalah penjumlahan dari setiap variabel *scale factors*.
- e. A adalah konstanta produktifitas. A = 2.94
- f. B = 0.91
- g. D = 0.28

Adapun 17 effort multiplier yang dimaksud adalah

1. *Required Software Reliability* (RELY): parameter ini berfungsi untuk mengukur tingkat kemampuan aplikasi melakukan fungsinya pada periode waktu tertentu. Gambar 2.1. menjelaskan setiap tingkatan parameter beserta nilai dan deskripsinya masing – masing.

RELY Descriptors	slight inconvenience	low, easily recoverable losses	moderate, easily recoverable losses	high financial loss	risk to human life	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	0.82	0.92	1.00	1.10	1.26	n/a

Gambar 2. 1. Deskripsi RELY

2. *Data Base Size (DATA)*: parameter ini berfungsi untuk menjelaskan besaran data yang disimpan pada penyimpanan yang tidak utama. D/P adalah besaran basis data dalam bytes atau karakter dibagi jumlah baris kode dalam SLOC. Gambar 2.2. menjelaskan setiap tingkatan parameter beserta nilai dan deskripsinya masing – masing.

DATA Descriptors		Testing DB bytes/Pgm SLOC < 10	10 = D/P < 100	100 = D/P < 1000	D/P = 1000	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	n/a	0.90	1.00	1.14	1.28	n/a

Gambar 2. 2. Deskripsi DATA

3. *Developed for Reusability (RUSE)*: parameter ini berfungsi untuk memperkirakan usaha yang dibutuhkan dalam rangka mengembangkan komponen yang akan digunakan kembali pada proyek berikutnya. Gambar 2.3. menjelaskan setiap tingkatan parameter beserta nilai dan deskripsinya masing – masing.

RUSE Descriptors		none	across project	across program	across product line	across multiple product lines
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	n/a	0.95	1.00	1.07	1.15	1.24

Gambar 2. 3. Deskripsi RUSE

4. *Documentation Match to Life-Cycle Needs (DOCU)*: parameter ini menjelaskan mengenai kesesuaian antara dokumentasi proyek dengan kebutuhan pengembangan perangkat lunak. Gambar 2.4. menjelaskan setiap tingkatan parameter beserta nilai dan deskripsinya masing – masing.

DOCU Descriptors	Many lifecycle needs uncovered	Some lifecycle needs uncovered.	Right-sized to life-cycle needs	Excessive for life-cycle needs	Very excessive for life-cycle needs	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	0.81	0.91	1.00	1.11	1.23	n/a

Gambar 2. 4. Deskripsi DOCU

5. *Execution Time Constraint (TIME)*: parameter ini menjelaskan persentasi ketepatan waktu eksekusi perintah yang diharapkan. Gambar 2.5. menjelaskan setiap tingkatan parameter beserta nilai dan deskripsinya masing – masing.

TIME Descriptors			= 50% use of available execution time	70% use of available execution time	85% use of available execution time	95% use of available execution time
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	n/a	n/a	1.00	1.11	1.29	1.63

Gambar 2. 5. Deskripsi TIME

6. *Main Storage Constraint (STOR)*: parameter ini merepresentasikan persentasi besaran penyimpanan yang digunakan. Gambar 2.6. menjelaskan setiap tingkatan parameter beserta nilai dan deskripsinya masing – masing.

STOR Descriptors			= 50% use of available storage	70% use of available storage	85% use of available storage	95% use of available storage
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	n/a	n/a	1.00	1.05	1.17	1.46

Gambar 2. 6. Deskripsi STOR

7. *Platform Volatility (PVOL)*: parameter ini menjelaskan kompleksitas perangkat lunak dan keras yang digunakan seperti sistem operasi, sistem manajemen basis data (DBMS), dll. Gambar 2.7 menjelaskan setiap tingkatan parameter beserta nilai dan deskripsinya masing – masing.

PVOL Descriptors		Major change every 12 mo.; Minor change every 1 mo.	Major: 6 mo.; Minor: 2 wk.	Major: 2 mo.; Minor: 1 wk.	Major: 2 wk.; Minor: 2 days	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	n/a	0.87	1.00	1.15	1.30	n/a

Gambar 2. 7. Deskripsi PVOL

8. *Analyst Capability (ACAP)*: parameter ini menjelaskan kemampuan dari analis dalam mengobservasi kebutuhan serta desain. Gambar 2.8 menjelaskan setiap tingkatan parameter beserta nilai dan deskripsinya masing – masing.

ACAP Descriptors	15th percentile	35th percentile	55th percentile	75th percentile	90th percentile	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.42	1.19	1.00	0.85	0.71	n/a

Gambar 2. 8. Deskripsi ACAP

9. *Programmer Capability* (PCAP): parameter ini menjelaskan kemampuan dari programmer sebagai sebuah tim yang mampu berkomunikasi dan bekerjasama. Gambar 2.9 menjelaskan setiap tingkatan parameter beserta nilai dan deskripsinya masing – masing.

PCAP Descriptors	15th percentile	35th percentile	55th percentile	75th percentile	90th percentile	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.34	1.15	1.00	0.88	0.76	n/a

Gambar 2. 9. Deskripsi PCAP

10. *Personnel Continuity* (PCON): parameter ini mendeskripsikan persentase pergantian pegawai dalam setahun. Gambar 2.10 menjelaskan setiap tingkatan parameter beserta nilai dan deskripsinya masing – masing.

PCON Descriptors	48% / year	24% / year	12% / year	6% / year	3% / year	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.29	1.12	1.00	0.90	0.81	

Gambar 2. 10. Deskripsi PCON

11. *Applications Experience* (APEX): parameter ini menjelaskan tingkat pengalaman tim pengembang dalam mengembangkan perangkat lunak. Gambar 2.11 menjelaskan setiap tingkatan parameter beserta nilai dan deskripsinya masing – masing.

APEX Descriptors	≤ 2 months	6 months	1 year	3 years	6 years	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.22	1.10	1.00	0.88	0.81	n/a

Gambar 2. 11. Deskripsi APEX

12. *Platform Experience* (PLEX): parameter ini menjelaskan tingkat pengalaman tim pengembang dalam mengembangkan antar muka, basis data, jaringan dan distribusi sistem. Gambar 2.12 menjelaskan setiap tingkatan parameter beserta nilai dan deskripsinya masing – masing.

PLEX Descriptors	≤ 2 months	6 months	1 year	3 years	6 year	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.19	1.09	1.00	0.91	0.85	n/a

Gambar 2. 12. Deskripsi PLEX

13. *Language and Tool Experience* (LTEX): parameter ini menjelaskan tingkat pengalaman tim pengembang dalam bahasa pemrograman dan penggunaan tool. Gambar 2.13 menjelaskan setiap tingkatan parameter beserta nilai dan deskripsinya masing – masing.

LTEX Descriptors	≤ 2 months	6 months	1 year	3 years	6 year	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.20	1.09	1.00	0.91	0.84	

Gambar 2. 13. Deskripsi LTEX

14. *Use of Software Tools* (TOOL): parameter ini menjelaskan kompleksitas tool yang digunakan dalam pengembangan. Gambar 2.14 menjelaskan setiap tingkatan parameter beserta nilai dan deskripsinya masing – masing.

TOOL Descriptors	edit, code, debug	simple, frontend, backend CASE, little integration	basic lifecycle tools, moderately integrated	strong, mature lifecycle tools, moderately integrated	strong, mature, proactive life-cycle tools, well integrated with processes, methods, reuse	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.17	1.09	1.00	0.90	0.78	n/a

Gambar 2. 14. Deskripsi TOOL

15. *Multisite Development* (SITE): parameter ini merepresentasikan dua hal yaitu distribusi data serta penunjang komunikasi. Gambar 2.15 menjelaskan setiap tingkatan parameter beserta nilai dan deskripsinya masing – masing.

Collocation Descriptors	International	Multi-city and Multi-company	Multi-city or Multi-company	Same city or metro. area	Same building or complex	Fully collocated
Communications Descriptors	Some phone, mail	Individual phone, FAX	Narrow band email	Wideband electronic communication.	Wideband elect. comm., occasional video conf.	Interactive multimedia
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.22	1.09	1.00	0.93	0.86	0.80

Gambar 2. 15. Deskripsi SITE

16. *Required Development Schedule* (SCED): parameter ini menjelaskan persentasi akselerasi penyelesaian proyek terhadap waktu yang diberikan. Gambar 2.16 menjelaskan setiap tingkatan parameter beserta nilai dan deskripsinya masing – masing.

SCED Descriptors	75% of nominal	85% of nominal	100% of nominal	130% of nominal	160% of nominal	
Rating Level	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multiplier	1.43	1.14	1.00	1.00	1.00	n/a

Gambar 2. 16. Deskripsi SCED

17. *Product Complexity* (CPLX): parameter ini dibagi ke dalam lima area yaitu operasi pengendalian, komputasional, kemandirian perangkat, manajemen data, manajemen antar muka. Gambar 2.17 menjelaskan setiap tingkatan parameter beserta nilai dan deskripsinya masing – masing.

Rating Level	Very Low	Low	Nominal	High	Very High	Extra High
EM	0.73	0.87	1.00	1.17	1.34	1.74

Gambar 2. 17. Deskripsi CPLX

Sedangkan kelima *scale factors* yang digunakan pada COCOMO II adalah :

1. *Precedentedness* (PREC): parameter ini menjelaskan pengalaman terdahulu terhadap proyek yang identik dengan proyek yang sedang dikembangkan. *Very Low* berarti tidak ada pengalaman sedangkan *Extra High* berarti sangat paham.
2. *Development Flexibility* (FLEX): parameter ini menjelaskan fleksibilitas dari proses pengembangan. *Very Low* berarti proses sesuai dengan yang tertulis sedangkan *Extra High* berarti proses sangat fleksibel
3. *Architecture/Risk Resolution* (RESL): parameter ini menjelaskan sejauh mana analisis terhadap resiko dilakukan. *Very Low* berarti sangat sedikit analisis sedangkan *Extra High* berarti analisis telah menyeluruh.
4. *Team Cohesion* (TEAM): parameter ini menjelaskan seberapa jauh anggota tim mengenal satu dengan yang lain. *Very Low* berarti sulit berinteraksi sedangkan *Extra High* berarti tidak ada permasalahan komunikasi.
5. *Process Maturity* (PMAT): parameter ini menjelaskan kematangan dari organisasi. *Very Low* berarti organisasi belum terstruktur *Extra High* berarti organisasi sudah mapan atau matang.

Gambar 2.18 menampilkan nilai dari setiap tingkatan yang terdapat pada *scale factors*.

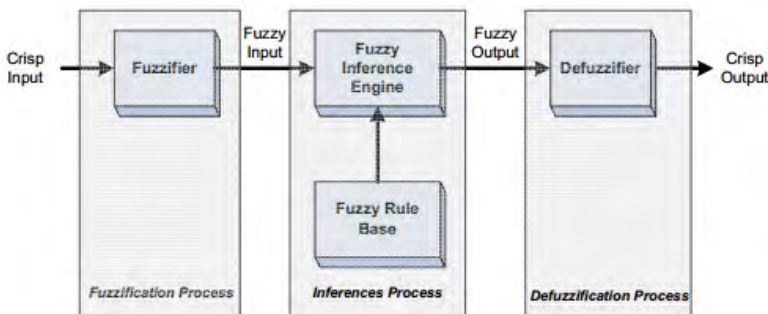
Drivers	Symbol		Scale Factors					
			Very Low	Low	Nominal	High	Very High	Extra High
Precedentedness	PREC	SF1	6.20	4.96	3.72	2.48	1.24	0.00
Development Flexibility	FLEX	SF2	5.07	4.05	3.04	2.03	1.01	0.00
Architecture/Risk Resolution	RESL	SF3	7.07	5.65	4.24	2.83	1.41	0.00
Team Cohesion	TEAM	SF4	5.48	4.38	3.29	2.19	1.10	0.00
Process Maturity	PMAT	SF5	7.80	6.24	4.68	3.12	1.56	0.00

Gambar 2. 18. Nilai dari *scale factors*

2.5. Fuzzy Logic

Fuzzy Logic System adalah satu dari tiga komponen utama dalam *soft computing*. Tujuan dari penggunaan *fuzzy logic* ialah untuk menangani ketidakpastian, ketidaktepatan, dan perkiraan untuk mencapai solusi yang diinginkan. Pada tahun 1965, Prof. Lofti Zadeh memperkenalkan *fuzzy logic* untuk menyelesaikan permasalahan yang sulit dipecahkan secara kuantitatif [14]. *Fuzzy logic system* menggunakan *fuzzy parameter* untuk mengatasi ketidakpastiaan dan ketidaktepatan dengan cara memetakan *input* ke *output* menggunakan *framework* yang disebut *Fuzzy Inference System* [15].

Framework ini terdiri dari tiga bagian utama yaitu *Fuzzification Process*, *Inferences Process* dan *Deffuzification Process*. Gambar 2.19. menampilkan diagram *fuzzy inference system* dengan tiga proses utama.

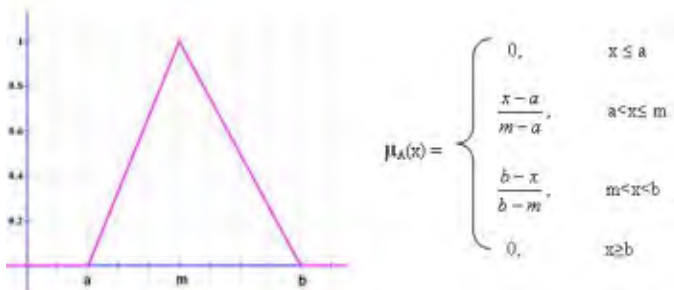


Gambar 2. 19. *Fuzzy Inference System*

2.5.1. Fuzzification Process

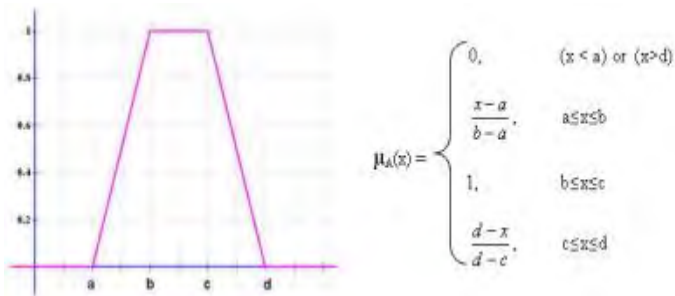
Fuzzification Process terdiri dari sebuah fuzzifier yang merubah *crisp input* menjadi nilai *fuzzy set* berdasarkan *membership function*. *Fuzzy set* adalah model matematika yang meliputi data yang bersifat kualitatif maupun kuantitatif, yang umumnya didapatkan dengan sarana bahasa natural. *Membership function* adalah sebuah kurva yang yang memetakan *input* menjadi *membership value* yang memiliki rentang antara 0 hingga 1. *Fuzzification process* memasukkan *input* kedalam sistem dengan menggunakan bahasa natural. *Membership function* pada *fuzzy system* memiliki beberapa tipe yaitu *triangular*, *trapezoidal*, dan *gaussian*.

- a. *Triangular Membership Function*: sebuah fungsi $\mu_A(x)$ yang bergantung pada parameter skalar a sebagai batas bawah, b sebagai batas atas, dan $a < m < b$. Gambar 2.20. menampilkan diagram dari *Triangular MF*.



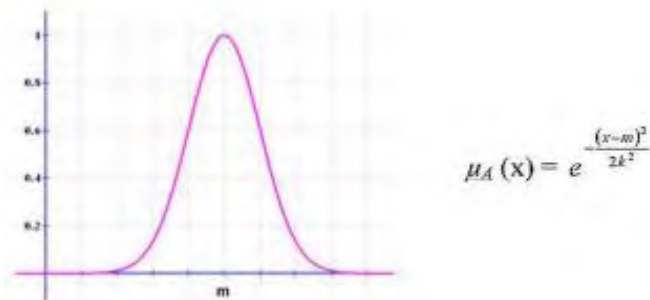
Gambar 2. 20. *Triangular Membership Function*

- b. *Trapezoidal Membership Function*: sebuah fungsi $\mu_A(x)$ yang bergantung pada parameter skalar a sebagai batas bawah, d sebagai batas atas, b sbagai batas tengah bawah, c sebagai batas tengah atas dan $a < b < c < b$. Gambar 2.21. menampilkan diagram dari *Trapezoidal MF*.



Gambar 2. 21. *Trapezoidal Membership Function*

- c. *Gaussian Membership Function*: sebuah fungsi $\mu_A(x)$ yang bergantung pada nilai tengah m dan dengan sebuah standar deviasi $k > 0$. Semakin kecil nilai k , semakin rapat kurva. Gambar 2.22. menampilkan diagram dari *Gaussian MF*.



Gambar 2. 22. *Gaussian Membership Function*

2.5.2. Inference Process

Inference process melibatkan *fuzzy inference engine* yang digunakan untuk memetakan antara *input* yang berasal dari *fuzzification process* dan *output* yang didapat dari pakar. Tugas dari *fuzzy rule* dalam *inference process* adalah menangkap ketidaktepatan dan bertindak sebagai sarana untuk menghasilkan

fuzzy output dari *fuzzy input*. *Fuzzy rule* dikenal juga dengan *Fuzzy IF-THEN rule* yang secara umum diekspresikan pada Persamaan 2.4.

$$\text{IF } (x \text{ is } A) \text{ AND } (y \text{ is } B) \text{ THEN } (z \text{ is } Z) \quad \dots(2.4)$$

Dimana:

- a. x, y, z merepresentasikan variabel.
- b. A, B, Z adalah nilai linguistik.

Fuzzy rule dapat dibagi dua bagian, bagian IF, yang digunakan sebagai premis yang mengandung deskripsi *fuzzy* untuk mengukur *input*, dan bagian THEN merupakan kesimpulan yang menentukan semua kemungkinan *output* dari setiap *input*. Pada proses ini, beberapa *fuzzy rule* dapat diaplikasikan pada *fuzzy inference engine* untuk menciptakan dasar pengetahuan yang digunakan dalam pengambilan keputusan dengan melakukan agregasi *fuzzy* untuk memetakan antara *input* ke *output*, yang berdasarkan modus ponens pada Persamaan 2.5.

$$\begin{array}{l} \text{Premis 1: IF } x \text{ is } A \text{ THEN } y \text{ is } B \\ \text{Premis 2: } x \text{ is } A' \end{array} \quad \dots(2.5)$$

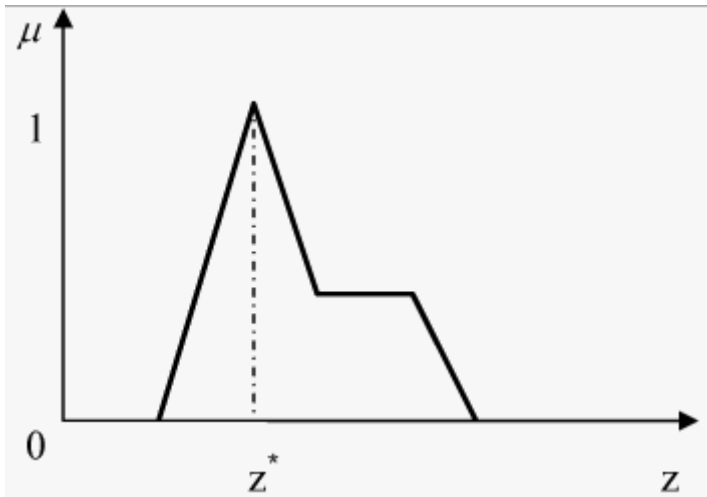
Kesimpulan: $y \text{ is } B'$

Variabel A, A', B, B' adalah *fuzzy set* dan x, y adalah simbol untuk objek. *Inference process* menciptakan *fuzzy output* sebagai agregasi dari beberapa *fuzzy rule*.

2.5.3. Defuzzification Process

Defuzzification process memproduksi dan menterjemahkan agregasi *fuzzy output* dari *inference process* menjadi hasil kuantitatif. Gambar 2.23. mengilustrasikan *defuzzification process* yang mengubah *fuzzy value* menjadi sebuah *crisp output*. Dimana μ merepresentasikan *Membership Function*

pada z . Nilai z^* merupakan titik puncak dari grafik μ yang merupakan titik yang nantinya akan berubah menjadi *crisp output*.



Gambar 2. 23. *Defuzzification Process*

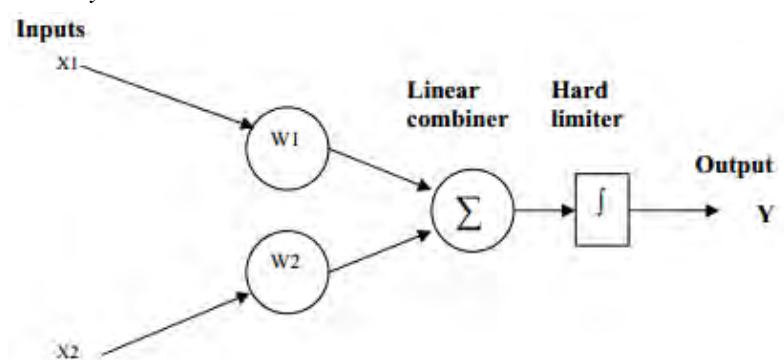
Dengan kemampuan meniru pikiran manusia dan mengatasi ketidakpastian, *fuzzy logic* system menjadi alat yang tepat untuk manajemen resiko yang terkait dengan ketidaktepatan hasil dalam peristiwa yang tidak pasti.

2.6. Neural Network

Artificial Neural Networks terinspirasi dari dasar pemikiran mengenai bagaimana otak bekerja. Otak terdiri dari begitu banyak sel saraf yang saling berhubungan, atau alat pengolah informasi, yang disebut neuron [16]. Setiap neuron adalah alat pengolah informasi. Untuk *membangun artificial neural network* (ANN), pertama kita harus menentukan berapa banyak jumlah neuron dan bagaimana neuron tersebut saling terhubung sehingga menciptakan suatu jaringan.

Secara umum, *nilai input* dikalikan dengan *weight*, dijumlahkan, lalu dikirim ke neuron yang lainnya lalu pada akhirnya ke neuron *output*. *Weight* menyesuaikan secara sistematis, bergantung pada data yang diberikan untuk mengoptimasi vektor *output*. Sebuah *neural network* yang belajar melalui penyesuaian *weight* yang berulang-ulang disebut dengan *training*.

Pada 1958, Frank Roseblatt memperkenalkan *training algorithm* yang menyediakan prosedur awal untuk melakukan *training* pada ANN disebut *The Perceptron*. *The perceptron* merupakan bentuk paling sederhana dari ANN yang terdiri dari sebuah neuron dengan *weight* yang dapat menyesuaikan dan fungsi lain. Gambar 2.24. menampilkan sebuah *perceptron* dengan 2 buah *input* dan 1 (*single*) *layer*. *Single layer perceptron* dapat mengklasifikasi hanya fungsi yang bersifat dapat dipisah secara linear, untuk fungsi yang dipisah tidak secara linear diperlukan *multilayer neural network*.



Gambar 2. 24. *Single-layer two-input perceptron*

Ada beberapa *learning algorithm* untuk melakukan *training* pada ANN, tapi metode yang paling populer adalah *back-propagation learning algorithm*. ANN dikenal dapat memberikan hasil yang baik ketika berhadapan dengan masalah yang kompleks antara hubungan *input* dan *output*, dan ketika *input data* memiliki nilai standar deviasi yang begitu beragam. Akan tetapi,

kemampuan dari *neural network* tergantung kepada arsitektur yang digunakan serta pengaturan pada parameternya.

BAB III

ANALISIS DAN PERANCANGAN SISTEM

Bab ini membahas tahap analisis permasalahan dan perancangan Tugas Akhir. Analisis permasalahan membahas bagaimana memodelkan data uji dengan menggunakan COCOMO dan bagaimana meningkatkan akurasi dari COCOMO. Sedangkan, perancangan membahas bagaimana membuat aplikasi berdasarkan analisa yang sudah dikembangkan.

3.1. Analisis

Dalam pengerjaan tugas akhir ini dilakukan analisa terhadap permasalahan yang ada. Analisa ini berfungsi untuk memodelkan permasalahan dengan metode yang ada serta meningkatkan akurasi metode tersebut.

3.1.1. Analisis Permasalahan

Permasalahan utama yang diangkat dalam pembuatan tugas akhir ini adalah bagaimana cara untuk mengoptimasi biaya dari sebuah perangkat lunak dengan menggunakan metode COCOMO. Sehingga dengan adanya optimasi ini kedepannya biaya perangkat lunak dapat diperkirakan dengan lebih akurat.

Untuk menunjang metode ini digunakan *dataset* yang didapat dari COCOMO 81 dan NASA 93. COCOMO 81 terdiri dari 63 data proyek perusahaan automotif dan kedirgantaraan yang mengembangkan perangkat lunak dalam menunjang perusahaan tersebut. Sedangkan NASA 93 merupakan *dataset* yang didapat dari proyek-proyek pengembangan perangkat lunak yang dilakukan oleh NASA sebanyak 93 proyek. Setiap data terdiri dari 17 variabel yaitu 15 variabel yang merepresentasikan *Effort Multiplier*, sebuah variabel KLOC dan satu buah variabel nilai yang sebenarnya. Dataset diatas diaplikasikan dengan COCOMO lalu hasilnya dibandingkan dengan nilai sebenarnya. Untuk memperkecil kesalahan atau ketidakakuratan antara nilai estimasi

dan nilai sebenarnya digunakan *fuzzy logic* dan *neural network*. Adapun langkah – langkah untuk menganalisa metode yang dikembangkan adalah observasi literatur terkait, perancangan COCOMO, memodelkan *fuzzy logic* ke dalam COCOMO, mengkonfigurasi *neural network* ke dalam COCOMO, dan evaluasi model.

3.1.1.1. Observasi Literatur Terkait

Dalam rangka memecahkan masalah estimasi biaya perangkat lunak, maka langkah awal yang dilakukan adalah studi literatur segala jenis bahan yang terkait. Langkah ini termasuk mengulas literatur mengenai memodelkan biaya perangkat lunak untuk mengenali setiap parameter yang ada didalamnya. Langkah ini dimulai dengan memahami penggunaan *cost drivers* yang terdapat dalam COCOMO, serta literatur lain yang terkait optimasi akurasi COCOMO.

3.1.1.2. Perancangan COCOMO

Setelah studi literatur dilakukan dan pemahaman akan COCOMO didapatkan, berikutnya dataset yang ada diimplementasikan ke dalam COCOMO. Hasil yang didapat berupa estimasi usaha yang digunakan selama pengembangan perangkat lunak. Kemudian nilai estimasi ini dibandingkan dengan nilai yang sebenarnya untuk menentukan apakah model yang digunakan sudah cukup akurat atau belum. Apabila model COCOMO belum akurat maka diimplementasikan metode lain untuk meningkatkan akurasi.

3.1.1.3. Memodelkan *Fuzzy Logic* ke dalam COCOMO

Dalam rangka meningkatkan akurasi, model COCOMO yang sebelumnya sudah dibuat dimodelkan kedalam *Fuzzy Logic*. *Fuzzy Logic* bertujuan untuk mengoptimasi nilai dari *setiap effort multiplier*. Cara mengoptimasi nilai tersebut adalah dengan

memodelkan *effort multiplier* ke dalam *fuzzy inference system*. *Membership Function* yang digunakan untuk memodelkan adalah *Gaussian Membership Function*. Dalam penelitian lain telah dibuktikan bahwa *Gaussian Membership Function* menghasilkan akurasi yang lebih baik dalam rangka estimasi biaya perangkat lunak dibandingkan dengan *Triangular* dan *Trapezoidal Membership Function* [17].

3.1.1.4. Mengkonfigurasi *Neural Network* ke COCOMO

Untuk lebih meningkatkan akurasi dari COCOMO, selain *fuzzy logic* digunakan metode *neural network*. Metode ini memetakan setiap *input* menjadi sebuah jaringan neuron yang terhubung dengan neuron lain di *layer* yang lain yang pada akhirnya menghasilkan sebuah *output*. Kemudian *output* tersebut dibandingkan dengan nilai sebenarnya. Apabila kesalahan masih belum bisa diterima maka dilakukan penyesuaian terhadap *weight* agar pada iterasi berikutnya hasilnya semakin dekat dengan nilai sebenarnya.

3.1.1.5. Evaluasi Model

Tahapan ini berfungsi untuk mengevaluasi semua model yang sudah dikembangkan. Tahapan ini juga membandingkan antara 1 model dengan model yang lainnya. Bukan hanya itu saja, apabila pada perkembangannya bisa diciptakan model baru maka akan dievaluasi apakah model tersebut lebih akurat dari model yang lainnya.

3.1.2. Data Uji

Terdapat dua buah *dataset* yang digunakan pada model yang dikembangkan yaitu COCOMO 81 dan NASA 91. Data tersebut digunakan secara umum pada penelitian yang menggunakan metode COCOMO.

3.1.2.1. COCOMO 81

Dataset ini merupakan *dataset* resmi yang dikembangkan sendiri untuk menciptakan COCOMO. *Dataset* ini berisi 63 *data points*. Data ini diambil dari proyek perusahaan yang bergerak di bidang kedirgantaraan yang bernama TRW dimana Barry W. Boehm bekerja sebagai direktur penelitian dan teknologi perangkat lunak. Setiap *data point* berisi 17 atribut yang terdiri dari 15 atribut COCOMO *effort multiplier*, 1 atribut baris kode (dalam ribuan) dan 1 atribut usaha yang sebenarnya. Keterangan atribut pada *dataset* dapat dilihat pada tabel 3.1.

Tabel 3. 1 Atribut COCOMO Dataset

Nomor	Atribut	Jenis
1	RELY	Numerik
2	DATA	Numerik
3	CPLX	Numerik
4	TIME	Numerik
5	STOR	Numerik
6	VIRT	Numerik
7	TURN	Numerik
8	ACAP	Numerik
9	AEXP	Numerik
10	PCAP	Numerik
11	VEXP	Numerik
12	LEXP	Numerik
13	MODP	Numerik
14	TOOL	Numerik
15	SCED	Numerik
16	LOC	Numerik
17	Actual Effort	Numerik

3.1.2.2. NASA 93

Dataset ini diambil dari proyek perangkat lunak yang dikembangkan oleh lembaga antariksa Amerika Serikat yang bernama NASA. *Dataset* ini diambil dari proyek tahun 1971 sampai dengan 1987. *Dataset* ini terdiri dari 93 data point. Setiap data point berisi 24 atribut yang terdiri dari 7 atribut yang mendeskripsikan proyek, 15 atribut COCOMO *effort multiplier*, satu atribut jumlah baris kode, satu atribut usaha yang sebenarnya. Keterangan atribut pada dataset dapat dilihat pada tabel 3.2.

Tabel 3. 2. Atribut NASA 93 Dataset

Nomor	Atribut	Jenis
1	Id	Numerik
2	Project Name	Alfabetis
3	Category of Application	Alfabetis
4	Flight or Ground System	Alfabetis
5	Nasa Center	Numerik
6	Years of Development	Numerik
7	Mode	Alfabetis
8	RELY	Alfabetis
9	DATA	Alfabetis
10	CPLX	Alfabetis
11	TIME	Alfabetis
12	STOR	Alfabetis
13	VIRT	Alfabetis
14	TURN	Alfabetis
15	ACAP	Alfabetis
16	AEXP	Alfabetis
17	PCAP	Alfabetis
18	VEXP	Alfabetis
19	LEXP	Alfabetis
20	MODP	Alfabetis
21	TOOL	Alfabetis
22	SCED	Alfabetis

Nomor	Atribut	Jenis
23	LOC	Numerik
24	Actual Effort	Numerik

3.1.3. Optimasi COCOMO dengan *Fuzzy Logic*

Hal yang penting ditekankan pada estimasi biaya perangkat lunak pada COCOMO adalah ketika ada ketidakpastian pada input di COCOMO model maka akan menyebabkan ketidakpastian pada *output* [18]. Maka dari itu diperlukan sebuah langkah atau penanganan yang substansial untuk menangani ketidakpastian ini.

Pendekatan *fuzzy logic* dalam estimasi biaya perangkat lunak adalah metode yang tepat ketika ketidakpastian pada informasi menjadi hal yang sangat diperhatikan. Menggunakan *fuzzy logic*, *cost drivers* dari sebuah proyek perangkat lunak dapat dispesifikasi berdasarkan distribusi nilai yang memungkinkan. *Cost drivers* merupakan variabel pemicu biaya suatu perangkat lunak. Secara umum, bentuk distribusi yang dapat merepresentasikan *cost drivers* adalah *fuzzy set*.

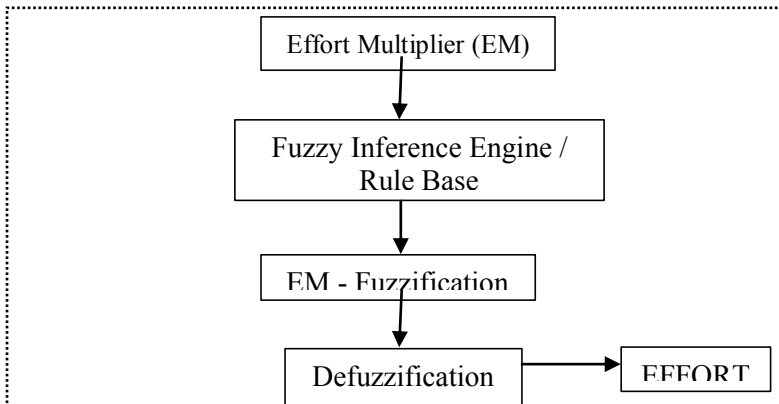
Daripada menggunakan nilai mutlak untuk mengkarakterisasi *cost drivers*, kita dapat menggunakan nilai interval yang diwakili oleh berbagai jenis *membership function* seperti *triangular*, *trapezoidal* dan *gaussian*. Pada literatur lain, sebuah jurnal menyimpulkan bahwa *gaussian membership function* memberikan akurasi yang lebih baik *dibanding triangular* dan *trapezoidal membership function* [1]. Pada *trapezoidal* ada kemungkinan bahwa beberapa atribut ditetapkan sebagai nilai maksimal ketika seharusnya nilai atribut tersebut lebih rendah, karena nilai maksimal pada *trapezoidal* memiliki rentang yang memungkinkan hal tersebut bisa terjadi.

Cost drivers sering kali diinterpretasikan dalam kategori yang tidak jelas yang membutuhkan penilaian secara subjektif. *Effort multiplier* dan *scale factors* pada COCOMO dideskripsikan menggunakan bahasa natural seperti *Very Low*, *Low*, *Nominal*,

High, Very High, Extra High dan direpresentasikan dengan nilai numerik [19]. Secara konvensional, estimasi biaya perangkat lunak dengan menggunakan metode COCOMO pada setiap *cost drivers*nya bergantung pada satu nilai untuk memprediksi usaha yang diperlukan. Dengan mengetahui hal di atas, kita dapat menggunakan menggunakan metode atau teknologi yang dapat mengevaluasi ketidaktepatan pada setiap *cost driver*. Dengan mengubah *cost drivers* kedalam *fuzzy set*, kita dapat meningkatkan akurasi estimasi biaya perangkat lunak.

3.1.4. Pendekatan Metode *Fuzzy* pada COCOMO

Dengan mempelajari sifat dari *cost driver* pada COCOMO, dipahami bahwa untuk meningkatkan akurasi pada estimasi perangkat lunak digunakan *gaussian membership function* untuk merepresentasikan *cost driver* yang berupa nilai linguistik. Input pada model ini yaitu berupa *effort multiplier* COCOMO. Dikarenakan dari semua *cost driver* pada COCOMO yaitu berupa *effort multiplier*, *scale factor*, dan LOC, dianalisa bahwa hanya *effort multiplier* yang memiliki deskripsi setiap atributnya berupa bahasa yang bisa diinterpretasikan menjadi *fuzzy set*. Arsitektur model ini bisa dilihat pada Gambar 3.1.



Gambar 3. 1. Arsitektur Fuzzy Logic untuk COCOMO

Dalam observasi yang dilakukan, tidak semua *effort multiplier* dapat direpresentasikan ke dalam *fuzzy set*. Hal ini dikarenakan deskripsi dari masing – masing *effort multiplier* dapat dikategorikan ke dalam dua tipe yaitu *kuantitatif dan kualitatif*. *Effort multiplier* yang kuantitatif adalah *effort multiplier* dimana deskripsinya bersifat mendeskripsikan suatu ukuran yang dapat diukur karena bersifat angka. Sedangkan *effort multiplier* yang bersifat kualitatif tidak dapat direpresentasikan karena deskripsinya bersifat kata-kata. Sebagai contoh adalah RELY, mengacu pada deskripsinya RELY dikatakan *Very High* apabila perangkat lunak itu dapat mengancam kehidupan manusia. Deskripsinya jelas sangat subjektif, artinya setiap orang akan berbeda-beda menafsirkannya. Sedangkan *effort multiplier* yang bersifat kuantitatif sebagai contoh adalah LTEX. LTEX dikatakan *Nominal* apabila tim sudah berpengalaman selama satu tahun menangani perangkat lunak yang menggunakan bahwa maupun tool yang sama dengan yang dikembangkan. Jelas hal ini dapat diukur tanpa adanya subjektifitas. Adapun *effort multiplier* yang bersifat kuantitatif adalah:

1. DATA
2. TIME
3. STOR
4. ACAP
5. PCAP
6. APEX
7. PLEX
8. LTEX
9. SCED
10. PVOL
11. PCON

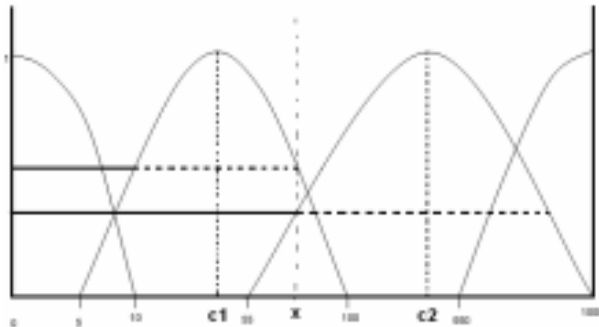
Dalam mengembangkan *fuzzy logic* ke dalam COCOMO terdapat 3 langkah yang harus dilakukan yaitu:

3.1.4.1. *Fuzzification*

Langkah yang pertama adalah *fuzzification*. *Fuzzification* mengubah input yang berupa *crisp data* ke dalam *fuzzy set*. Kesebelas *effort multiplier* yang bersifat kuantitatif dijadikan input untuk dilakukan *fuzzification*. Dalam tugas akhir ini menggunakan *gaussian membership function* sebagai model *fuzzy set* untuk setiap *effort multiplier*. Fungsi *Gaussian* direpresentasikan dalam Persamaan 3.1. [20].

$$\mu_{Ai}(x) = \text{Gaussian}(x, c_i, \sigma_i) = e^{-(x-c_i)^2/2\sigma_i^2} \quad \dots(3.1.)$$

Dimana c_i adalah nilai tengah dari *fuzzy set* ke- i dan σ_i adalah lebar dari kurva pada *fuzzy set* ke- i . Sebagai contoh, Gambar 3.2. merepresentasikan *effort multiplier* DATA kedalam *fuzzy set* dengan menggunakan *gaussian membership function*.



Gambar 3. 2. Representasi DATA menggunakan *Gaussian Membership Function*

3.1.4.2. *Fuzzy Inference Engine*

Fuzzy inference engine bertugas untuk membuat *fuzzy rules* pada *fuzzy set*. *Fuzzy rule* berisi variabel linguistik atau verbal yang berhubungan dengan proyek. *Fuzzy rule* ini berfungsi untuk memetakan antara *input* pada *fuzzy* dan *ouput* pada *fuzzy set*.

Persamaan 3.2. mengilustrasikan *fuzzy rules* pada COCOMO DATA *effort multiplier*. DATA merupakan *effort multiplier* yang dapat diganti dengan *effort multiplier* yang lain. Jumlah *fuzzy rules* pada setiap *effort multiplier* bergantung pada jumlah *membership function* yang ada. *Low* mendeskripsikan nilai linguistik yang terdapat pada *input* sedangkan *decreased* merepresentasikan nilai yang terdapat pada *output*.

R_1 : IF Input DATA is low THEN Output data is decreased

R_2 : IF Input DATA is nominal THEN Output data is unchanged

dan seterusnya... (3.2.)

3.1.4.3. Defuzzification

Langkah yang ketiga adalah *defuzzification*. *Defuzzification* mengubah *fuzzy output* menjadi *crisp data*. Adapaun *defuzzification* ditampilkan pada Persamaan 3.3.

$$y = \frac{\sum_{i=1}^K \mu(x_i)x_i}{\sum_{i=1}^K \mu(x_i)} \quad \dots(3.3.)$$

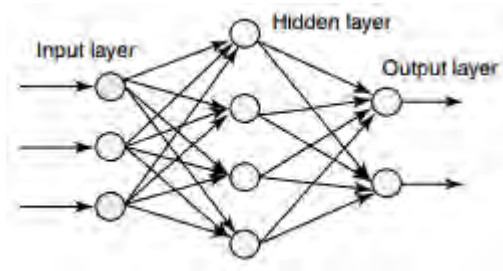
Dimana:

- y adalah *centre of gravity* (COG) yang merupakan nilai tengah dari *output* yang kemudian dirubah menjadi *crisp data*.
- K adalah jumlah dari *fuzzy set* dalam sebuah *fuzzy logic*.

3.1.5. Optimasi COCOMO dengan Neural Network

Subbab ini menjelaskan mengenai bagaimana meningkatkan akurasi dari COCOMO dengan menggunakan *neural network*. Kemampuan dari sebuah *neural network* ditentukan oleh arsitektur dan parameter lain yang digunakan. Ada banyak parameter yang terkait termasuk jumlah *layer*, jumlah *node* atau neuron pada setiap *layer*, *weight* antara dua buah *node* dan lain sebagainya. Sejauh ini teknik yang paling tepat dalam mengoptimasi estimasi biaya perangkat lunak adalah *multilayer*

feed forward dengan *back-propagation learning algorithm* dan aktivasi sigmoid. Gambar 3.3. menunjukkan *arsitektur multilyer neural network* secara umum. Dalam observasi yang dilakukan,

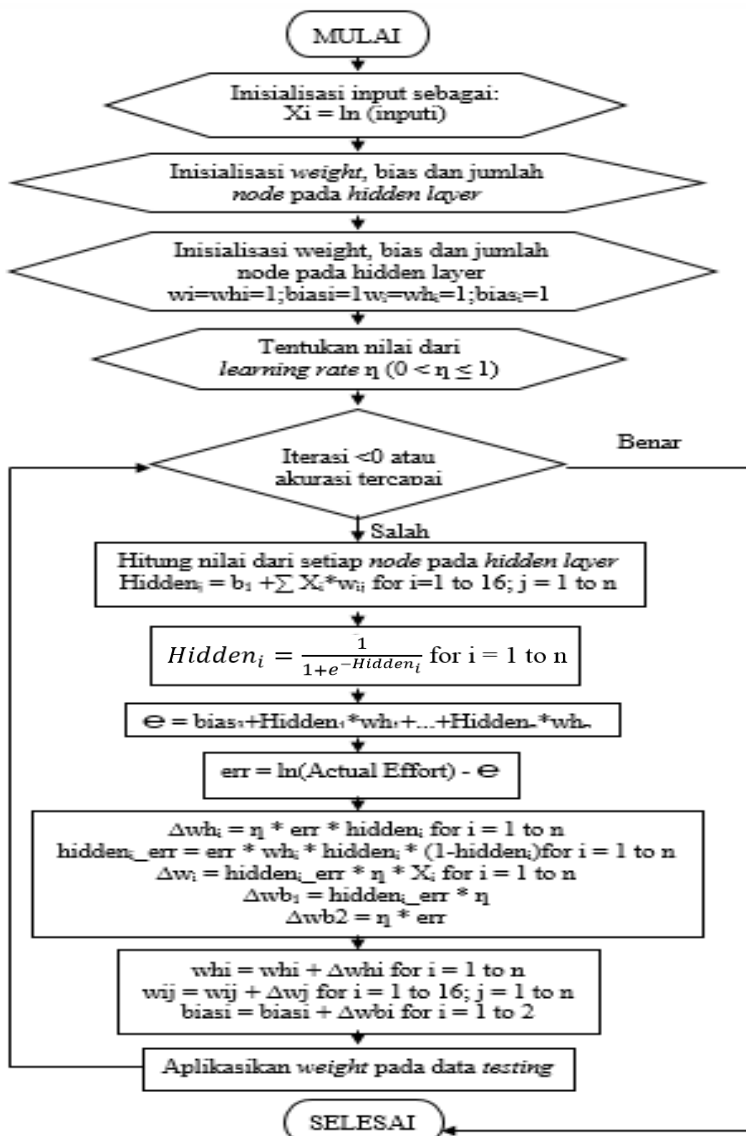


Gambar 3. 3. *Multilayer Neural Network*

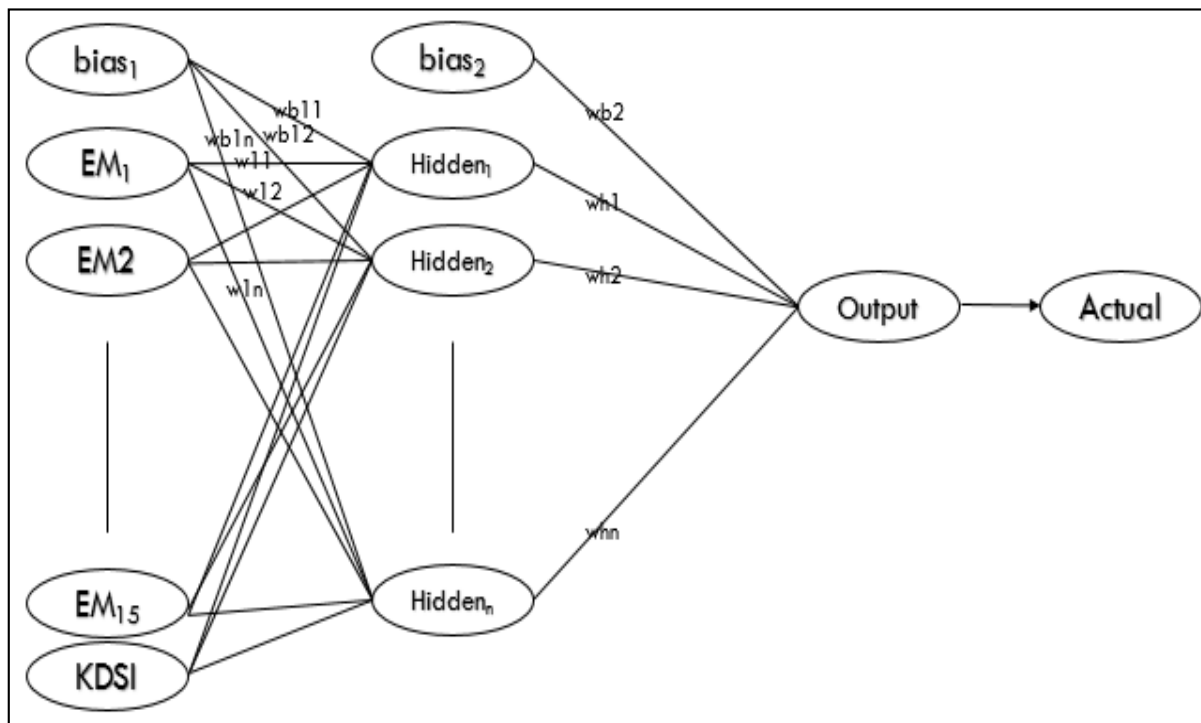
terdapat dua model arsitektur yang dikembangkan pada tugas akhir ini dalam rangka meningkatkan akurasi estimasi biaya proyek.

3.1.5.1. Basic Neural Network Architecture

Arsitektur ini merupakan pemodelan yang sederhana atau mendasar dengan menggunakan *neural network* pada COCOMO. Arsitektur ini menggunakan parameter pada COCOMO 81 sebagai *input*. Model ini memiliki 16 paramter *input* dan dua buah bias. 16 *input* tersebut terdiri dari 15 variabel *effort multiplier* dan 1 variable KDSI (jumlah baris kode). Terdapat sebuah *hidden layer* diantara *input* dan *output layer*. Tidak ada nilai yang mutlak untuk jumlah *node* pada *hidden layer*. Arsitektur ini juga menggunakan aktivasi sigmoid untuk menghitung nilai dari setiap *node* pada *hidden layer*. Secara umum arsitektur ini dapat dilihat pada Gambar 3.5. Arsitektur ini juga diimplementasikan pada COCOMO yang sudah mengadopsi fuzzy logic. Secara umum langkah-langkah untuk mengimplementasikan arsitektur ini dapat dilihat pada Gambar 3.4.



Gambar 3.4 . Diagram Alur Implementasi Basic Architecture



Gambar 3. 5. Arsitektur *Basic Neural Network*

3.1.5.2. *Modified Neural Network Architecture*

Pada arsitektur ini, perubahan yang signifikan dengan arsitektur yang sebelumnya adalah penggunaan *Scale Fator* yang diambil dari COCOMO II. Model ini memiliki 22 parameter *input* dan tiga buah bias. 22 input tersebut terdiri dari 15 variabel *effort multiplier*, 5 variable *scale factor*, dan 1 variable KLOC (jumlah baris kode). Antara *input* dan *output layer* terdapat dua buah *node* pada *hidden layer*. Namun berbeda dengan arsitektur sebelumnya, pada arsitektur ini tidak semua *input* terhubung ke setiap *node* pada *hidden layer*. *Hidden layer* 1 menampung *input* dari *effort multiplier* sedangkan *hidden layer* 2 menampung *input* dari *scale factor* dimana *weight* antara *scale factor* dan *hidden layer* 2 ditambah dengan $\ln(\text{size})$. Arsitektur ini juga menggunakan aktivasi sigmoid untuk menghitung nilai dari setiap *node* pada *hidden layer*. Secara umum arsitektur ini dapat dilihat pada Gambar 3.7. Pada arsitektur ini, setiap input diubah ke dalam bentuk logaritma natural seperti berikut:

$$\ln(PM) = \ln(A) + \ln(EM_1) + \ln(EM_2) + \dots + \ln(EM_{15}) \\ + [1.01 + SF_1 + \dots + SF_5] * \ln(\text{size})$$

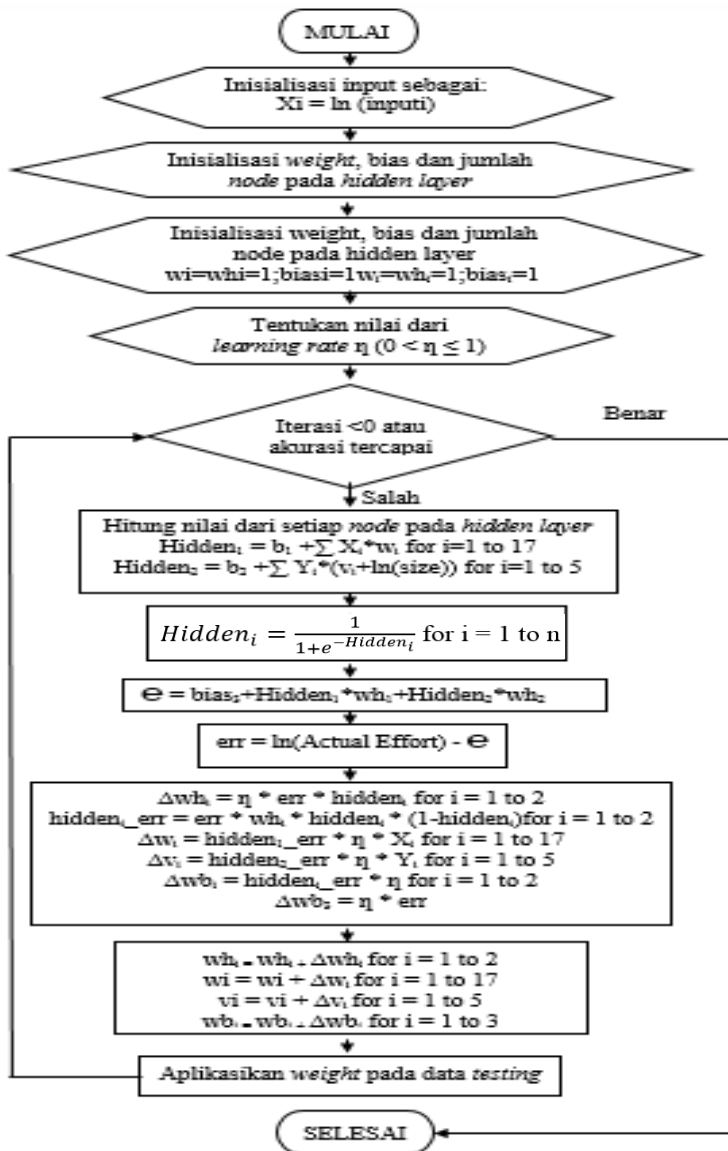
Persamaan diatas dirubah menjadi:

$$E = [b_1 + w_1 * X_1 + w_2 * X_2 + \dots + w_{15} * w_{15}] \\ + [b_2 + Y_1 + \dots + Y_5] * [v_1 + \ln(\text{size})]$$

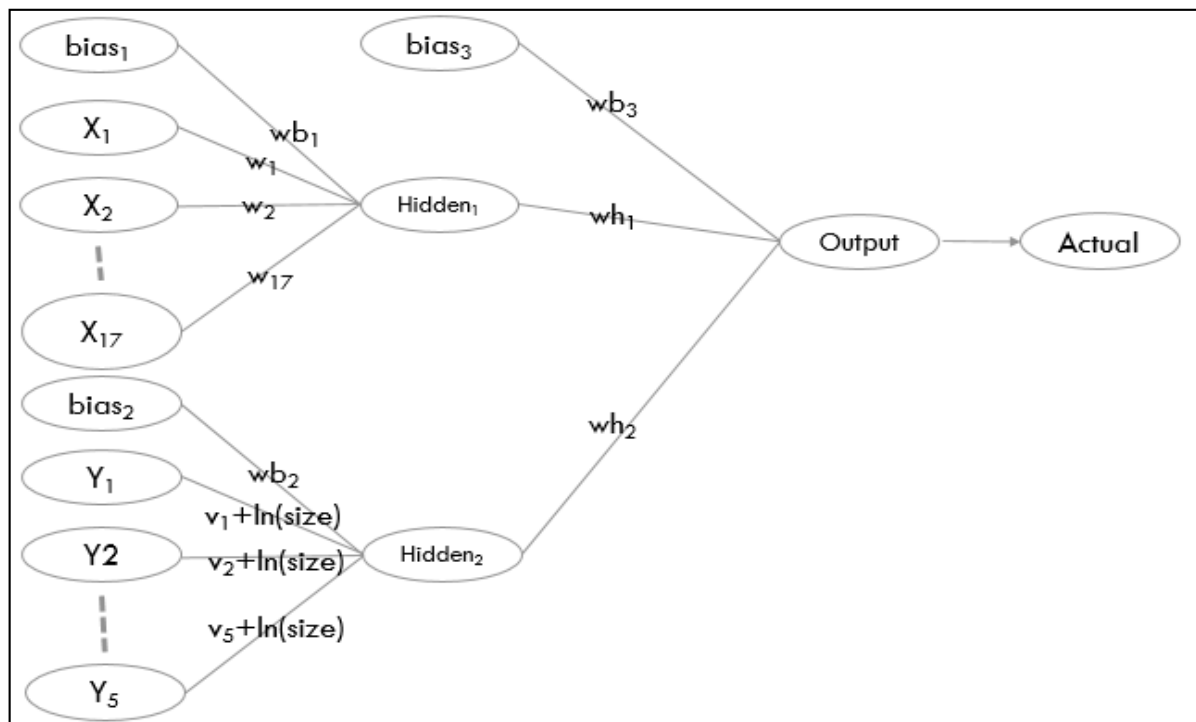
Dimana:

- a. $E = \ln(PM)$
- b. $X_1 = \ln(EM_1); \dots; X_{17} = \ln(EM_{17})$
- c. $Y_1 = SF_1; \dots; Y_5 = SF_5$

Arsitektur ini juga diimplementasikan pada COCOMO yang sudah mengadopsi *fuzzy logic*. Secara umum langkah-langkah untuk mengimplementasikan arsitektur dapat dilihat pada Gambar 3.6.



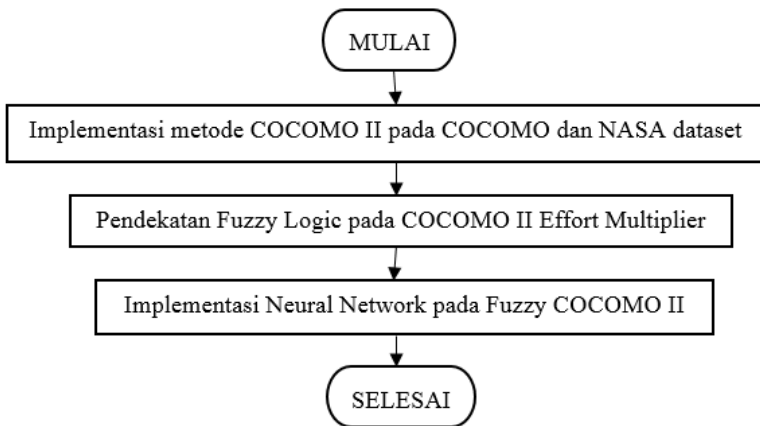
Gambar 3. 6. Diagram Alur Implementasi *Modified Architecture*



Gambar 3. 7. Arsitektur *Modified Neural Network*

3.1.6. Pendekatan *Fuzzy* dan *Artificial Neural Network* pada COCOMO

Pada tugas akhir ini, akurasi dari COCOMO ditingkatkan dengan cara menggabungkan metode *Fuzzy* dan *Artificial Neural Network*. Gambar 3.8. menunjukkan proses ini secara umum.



Gambar 3. 8. Diagram Alur Proses Implementasi Fuzy Logic dan Neural Network pada COCOMO II

3.1.7. Deskripsi Umum Aplikasi berbasis Metode yang Dikembangkan

Selain menganalisa metode yang dapat meningkatkan akurasi estimasi perangkat lunak, tugas akhir ini juga memberikan kontribusi tambahan berupa perangkat lunak yang dapat menghitung estimasi biaya perangkat lunak berdasarkan metode optimasi yang dikembangkan. Aplikasi ini menghitung estimasi biaya perangkat lunak dengan memasukkan data *effort multiplier*, *scale factor*, dan LOC.

3.1.7.1. Spesifikasi Kebutuhan Perangkat Lunak

Bagian ini berisi semua kebutuhan perangkat lunak yang diuraikan secara rinci dalam bentuk diagram kasus, diagram urutan, dan diagram aktivitas. Masing-masing diagram menjelaskan perilaku atau sifat dari sistem ini. Kebutuhan perangkat lunak dalam sistem ini mencakup kebutuhan fungsional saja. Pada bab ini juga dijelaskan tentang spesifikasi terperinci pada masing-masing kebutuhan fungsional. Rincian spesifikasi dari kasus penggunaan disajikan dalam bentuk tabel.

3.1.7.2. Kebutuhan Fungsional

Kebutuhan fungsional berisi kebutuhan utama yang harus dipenuhi oleh sistem agar dapat bekerja dengan baik. Kebutuhan fungsional mendefinisikan layanan yang harus disediakan oleh sistem, bagaimana reaksi terhadap masukan, dan apa yang harus dilakukan sistem pada situasi khusus. Daftar kebutuhan fungsional dapat dilihat pada Tabel 3.3.

Tabel 3. 3. Daftar Kebutuhan Fungsional Perangkat Lunak

Kode Kebutuhan	Kebutuhan Fungsional	Deskripsi
A0001	Memasukkan nilai atribut	Pengguna memasukkan atribut yang diperlukan seperti <i>effort multiplier</i> , <i>scale factor</i> , dan LOC.
A0002	Melihat hasil estimasi biaya perangkat lunak	Pengguna dapat melihat hasil estimasi biaya perangkat lunak

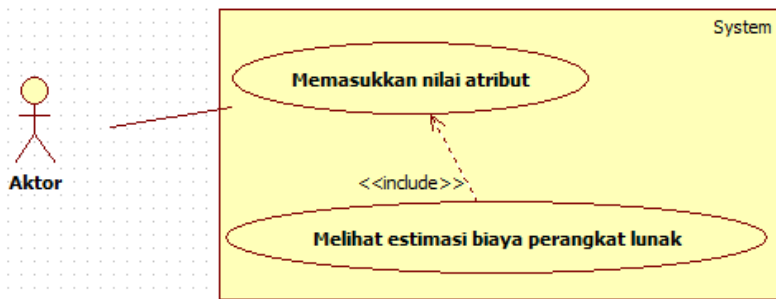
3.1.7.3. Aktor

Aktor mendefinisikan entitas-entitas yang terlibat dan berinteraksi langsung dengan sistem. Entitas ini bisa berupa manusia maupun sistem atau perangkat lunak yang lain. Penulis

mendefinisikan aktor untuk sistem ini yaitu pengguna dari aplikasi estimasi biaya perangkat lunak.

3.1.7.4. Kasus Pengguna

Kasus-kasus penggunaan dalam sistem ini akan dijelaskan secara rinci pada subbab ini. Kasus penggunaan secara umum akan digambarkan oleh salah satu model UML, yaitu diagram kasus penggunaan. Rincian kasus penggunaan berisi spesifikasi kasus penggunaan dan diagram aktivitas. Diagram kasus penggunaan dapat dilihat pada Gambar 3.9. Daftar kode diagram kasus penggunaan sistem dapat dilihat pada Tabel 3.4.



Gambar 3. 9. Diagram Kasus Penggunaan Sistem

Tabel 3. 4. Daftar Kode Diagram Kasus Penggunaan

Kode Kasus Pengguna	Nama
UC01	Memasukkan nilai atribut
UC02	Melihat estimasi biaya perangkat lunak

3.4.3.1. Memasukkan nilai atribut

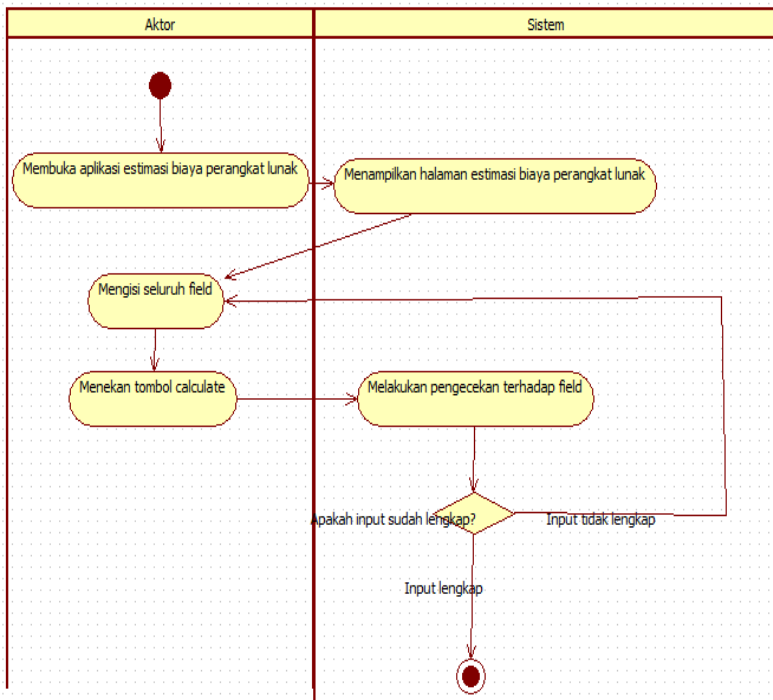
Pada kasus pengguna ini, pengguna memasukkan *effort multiplier*, *scale factor* dan LOC pada halaman aplikasi. Diagram

aktivitas dan diagram urutan dari kasus penggunaan ini bisa dilihat pada Tabel 3.5. dan Gambar 3.10.

Tabel 3. 5. Spesifikasi Kasus Pengguna Memasukkan Nilai Atribut

Nama	Memasukkan Nilai Atribut	
Kodel	UC01	
Deskripsi	Memasukkan nilai atribut berupa effort multiplier, scale factors dan LOC ke dalam sistem.	
Aktor	Pengguna	
Kondisi Awal	Field masih kosong.	
Alir Kejadian Normal	Pengguna	Sistem
	1. Pengguna membuka aplikasi estimasi perangkat lunak. 3. Pengguna mengisi seluruh <i>field</i> yang terdapat pada halaman. 4. Pengguna menekan tombol <i>calculate</i> .	2. Sistem menampilkan halaman estimasi biaya perangkat lunak. 5. Sistem melakukan pengecekan terhadap seluruh <i>field</i> yang diisi pengguna A.1. <i>Input</i> yang diberikan pengguna tidak lengkap atau tidak sesuai dengan ketentuan
Aliran Kejadian Alternatif	A1. <i>Input</i> yang diberikan pengguna tidak sesuai dengan ketentuan atau tidak lengkap. A1.1. Sistem menampilkan peringatan kepada pengguna untuk melengkapi <i>field</i> A1.2. Lanjut ke aliran normal 3	

Kondisi Akhir	Proses memasukkan atribut berhasil.
Kebutuhan Khusus	-



Gambar 3. 10. Diagram Aktivitas Menghitung Biaya Perangkat Lunak

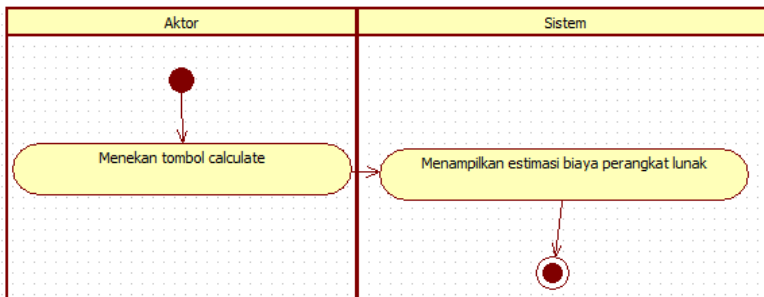
3.4.3.2. Melihat Estimasi Biaya Perangkat Lunak

Pada kasus pengguna ini, pengguna dapat melihat hasil estimasi biaya perangkat lunak. Kasus penggunaan ini hanya dapat terjadi apabila kasus penggunaan memasukkan atribut sudah

dilakukan. Diagram aktivitas dan diagram urutan dari kasus penggunaan ini bisa dilihat pada Tabel 3.6. dan Gambar 3.11.

Tabel 3. 6. Spesifikasi Kasus Pengguna Melihat Estimasi

Nama	Melihat Estimasi Biaya Perangkat Lunak	
Kodel	UC02	
Deskripsi	Melihat estimasi biaya perangkat lunak	
Aktor	Pengguna	
Kondisi Awal	Field sudah terisi	
Alir Kejadian Normal	Pengguna	Sistem
	1.Pengguna menekan tombol <i>calculate</i> .	2.Sistem menampilkan hasil biaya perangkat lunak
Aliran Kejadian Alternatif	-	
Kondisi Akhir	Hasil estimasi biaya perangkat lunak ditampilkan	
Kebutuhan Khusus	-	



Gambar 3. 11. Diagram Aktifitas Menampilkan estimasi biaya perangkat lunak

3.2. Perancangan Sistem

Penjelasan tahap perancangan perangkat lunak dijelaskan dalam bentuk perancangan antarmuka.

3.2.1. Perancangan Antarmuka Pengguna

Penjelasan tahap perancangan perangkat lunak dijelaskan dalam bentuk perancangan antarmuka.

3.2.2.1 Halaman Tampilan Menginput dan Menampilkan Estimasi Biaya Perangkat Lunak

Halaman ini merupakan halaman menghitung estimasi biaya perangkat lunak. Field yang disediakan berupa 15 effort multiplier, 5 scale factor, LOC dan biaya gaji karyawan perbulan. Gambar 3.12. menampilkan halaman estimasi biaya perangkat lunak.

The screenshot shows a window titled "Software Cost Estimation" with a standard Windows-style title bar (minimize, maximize, close buttons). The interface is organized into a grid of input fields. Each field consists of a text label followed by a dropdown menu currently set to "Nominal". The fields are arranged in three columns: Column 1: Analyst Capability, Modern Programming Practices, Language Experience, Database Size, Machine Volatility, Precededness, Development Reliability. Column 2: Programmers Capability, Use of Software Tools, Schedule Costraint, Time, Complexity, Process Maturity, Team. Column 3: Application Experiece, Virtual Machine, Main Memory, Turn, Reliability, Resolution. Below the grid, there are three input fields: "KLOC", "Cost", and "Total Waktu". To the right of the "Cost" field is a "Calculate" button. At the bottom, there are three more input fields: "Total Pekerja", "Estimasi Biaya", and an empty field for the result.

Analyst Capability:	Nominal	Programmers Capability:	Nominal	Application Experiece	Nominal
Modern Programming Practices	Nominal	Use of Software Tools:	Nominal	Virtual Machine:	Nominal
Language Experience:	Nominal	Schedule Costraint:	Nominal	Main Memory:	Nominal
Database Size:	Nominal	Time:	Nominal	Turn:	Nominal
Machine Volatility:	Nominal	Complexity:	Nominal	Reliability:	Nominal
Precededness:	Nominal	Process Maturity:	Nominal	Resolution:	Nominal
Development Reliability	Nominal	Team:	Nominal		

KLOC Cost

Total Pekerja Total Waktu Estimasi Biaya

Gambar 3. 12. Antarmuka Aplikasi Estimasi Biaya Perangkat Lunak

BAB IV IMPLEMENTASI

Setelah pada Bab IV dijelaskan mengenai analisa serta metode yang digunakan dalam rangka optimasi biaya perangkat lunak dengan menggunakan COCOMO, *fuzzy* dan *neural network*, maka pada Bab IV ini akan dijelaskan tahap implementasi dari metode yang dianalisa.

4.1. Lingkungan Implementasi

Implementasi dilakukan pada lingkungan sebagai berikut:

- a. Perangkat Keras
Implementasi dilakukan pada sebuah laptop dengan spesifikasi:
 - i. *Processsor* : Intel® Core™ i5-2410M Processor 2.30GHz with Turbo Boost up to 2.90GHz
 - ii. RAM : 4 GB
 - iii. HDD : 500 GB
- b. Perangkat Lunak
Implementasi dilakukan dengan perangkat lunak sebagai berikut:
 - i. *Operating System* : Windows 8.1 64-bit
 - ii. Implementasi COCOMO : Microsoft Excel
 - iii. Implementasi *Fuzzy* : Matlab R2013a
 - iv. Implementasi *Neural Network* : Dev-C++
 - v. Implementasi Aplikasi : Visual Studio 2012 64-bit

4.2. Penjelasan Implementasi

Seperti yang sudah dijelaskan pada bab analisis dan perancangan, pada bab implementasi ini ada beberapa hal yang akan diaplikasikan. Berikut metode atau desain yang akan diimplementasikan.

1. Implementasi *Fuzzy* pada COCOMO II
2. Implementasi *Neural Network* pada *Fuzzy* COCOMO II

3. Implementasi Aplikasi Estimasi Biaya Perangkat Lunak

4.3. Implementasi *Fuzzy* pada COCOMO II

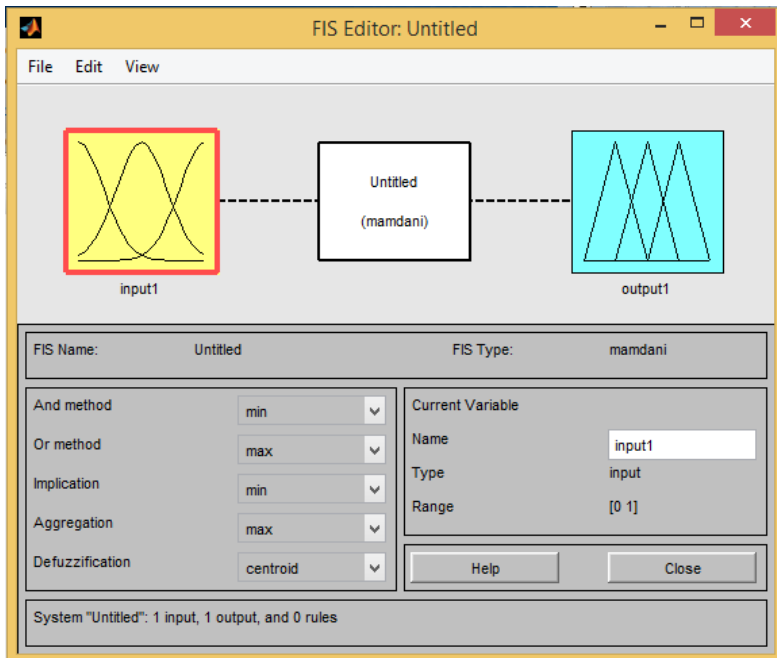
Implementasi ini dilakukan dengan cara memodelkan *effort multiplier* yang berupa *crisp data* ke dalam *fuzzy set* dengan *gaussian membership function* pada Matlab. Pada matlab telah tersedia sebuah *tool* yang disebut FIS. Setelah *input* dan *ouput* dimasukkan dan *rule* ditetapkan maka hasilnya diubah kembali menjadi *crisp data* untuk dipergunakan dalam rangka optimasi estimasi biaya perangkat lunak. Pada Lampiran C dapat dilihat hasil *fuzzy set* pada COCOMO dengan menggunakan FIS pada Matlab untuk beberapa *effort multiplier*. Berikut ini langkah – langkah implementasi Fuzzy Logic pada COCOMO II menggunakan Matlab :

i. Membuka *Fuzzy Inference System* pada Matlab

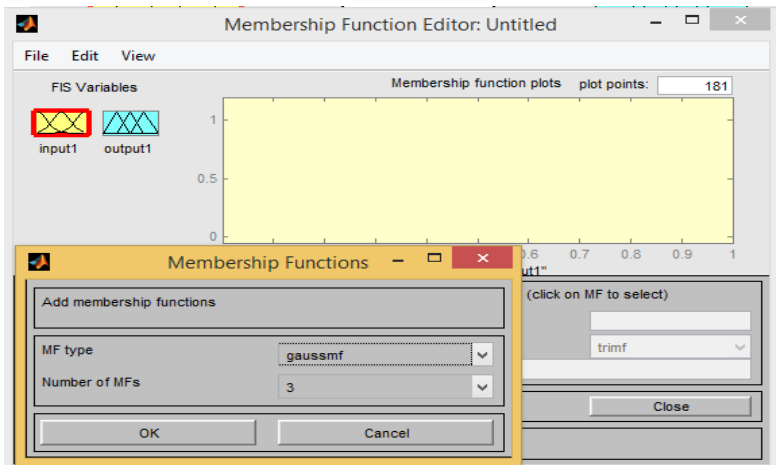
Pada implementasi *Fuzzy Logic* digunakan sebuah alat bantu yang disebut *Fuzzy Inference System Editor*. Alat bantu ini terdapat pada aplikasi Matlab. Gambar 4.1. menampilkan *FIS Editor* pada Matlab.

ii. Mengisi Jumlah *Membership Function* pada *FIS Editor*

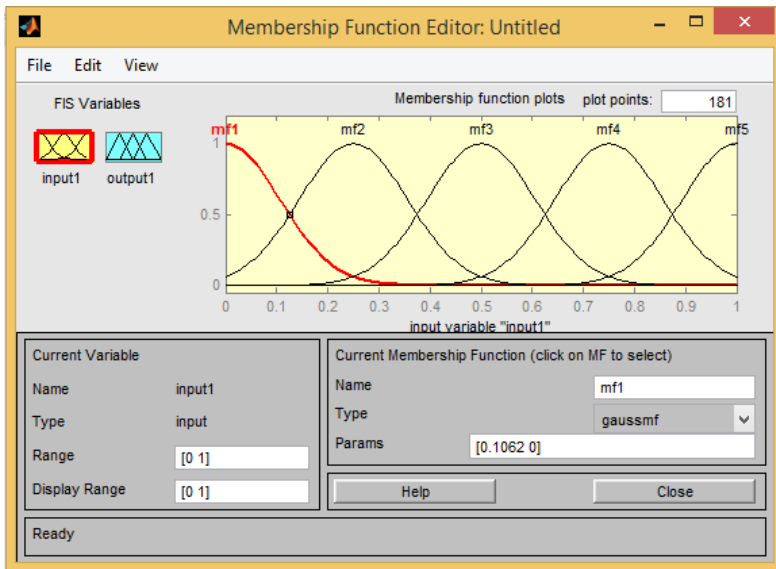
Input baru dibuat dengan cara mengisi pada *input* yang terdapat di *FIS Editor*. MF type dapat dipilih untuk menentukan tipe *membership function* dalam hal ini digunakan *Gaussian Membership Function*. Gambar 4.2. menampilkan *Membership Function* pada *FIS Editor*. Gambar 4.3. menampilkan *Membership Function Editor* pada FIS.



Gambar 4. 1. FIS Editor



Gambar 4.2. Membership Function pada FIS Editor



Gambar 4.3. *Membership Function Editor*

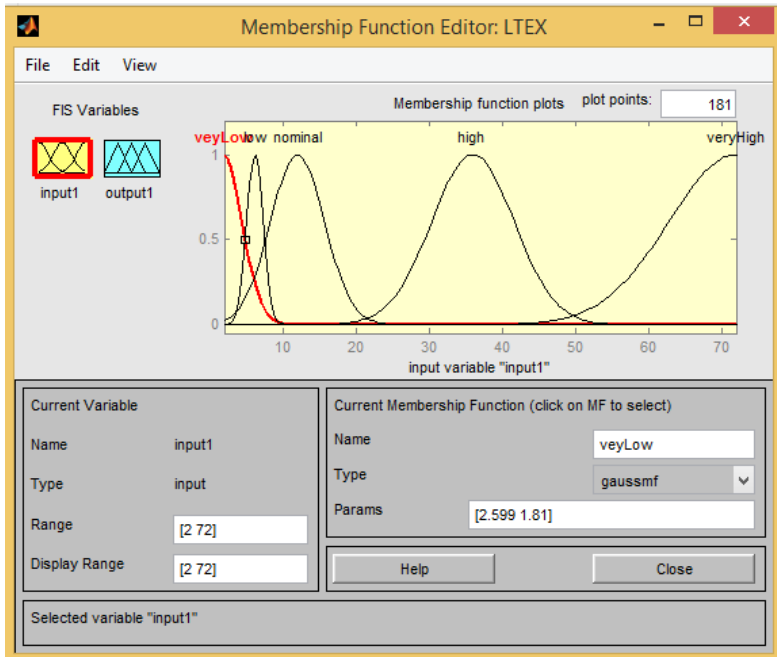
iii. Mengisi *Membership Function Editor*

Membership Function diisi berdasarkan keterangan yang terdapat pada setiap *effort multiplier*. Pada contoh berikut menggunakan sebuah *effort multiplier* yaitu LTEX. Pada setiap *effort multiplier* terdapat *descriptor* yang berfungsi menjelaskan *level* setiap *effort multiplier*.

Gambar 4.4. menampilkan LTEX *Descriptor* yang memetakan setiap *level* pada LTEX dengan deskripsi dalam tahun atau bulan. *Descriptor* pada LTEX memiliki rentang dari 2 bulan hingga 6 tahun. Deskripsi ini disamakan dalam bentuk bulan supaya dapat digunakan, sehingga pada *Membership Function Editor* diisi *Range* [2, 72]. Kemudian setiap *membership function* diisi berdasar LTEX *Descriptor*. Gambar 4.5. menunjukkan implementasi *Membership Function* pada LTEX.

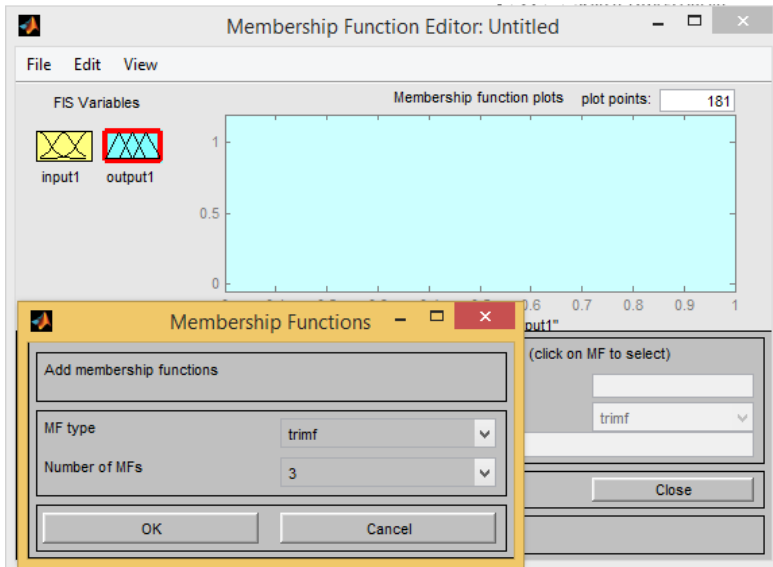
LTEX Descriptors	≤ 2 months	6 months	1 year	3 years	6 year
Rating Levels	Very Low	Low	Nominal	High	Very High

Gambar 4. 4. Deskripsi LTEX



Gambar 4. 5. LTEX Membership Function

- iv. Mengisi Jumlah *Output* pada *Membership Function Editor*
- Output* baru dibuat dengan cara mengisi pada *output* yang terdapat di *FIS Editor*. MF type dapat dipilih untuk menentukan tipe *membership function* dalam hal ini digunakan *Gaussian Membership Function*. Gambar 4.6. menampilkan *Membership Function* pada *FIS Editor*. Gambar 4.6. menampilkan *Output Membership Function Editor* pada *FIS*.



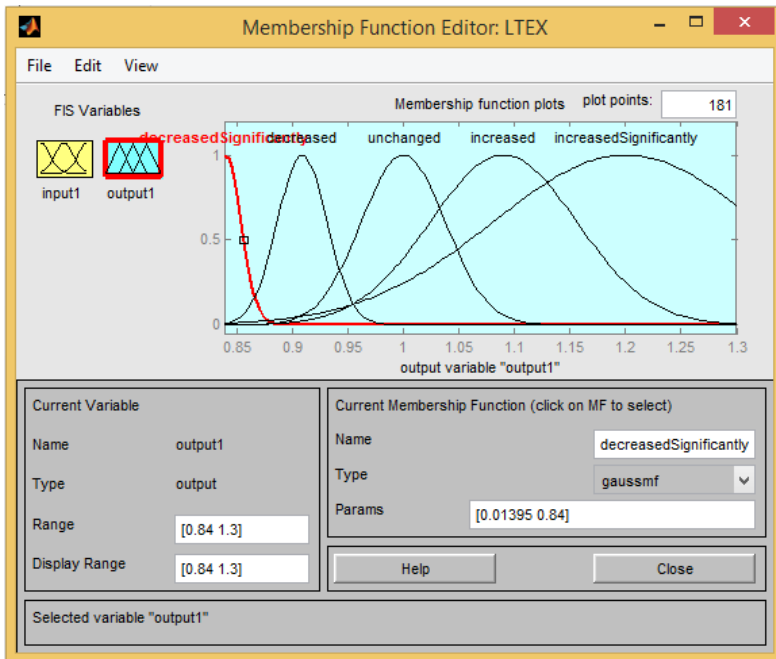
Gambar 4. 6. *Output pada Membership Function Editor*

- v. Mengisi *Output* pada *Membership Function Editor*

Pada *output*, *membership function* diisi berdasarkan nilai yang terdapat pada setiap *Effort Multiplier*. Gambar 4.7. menampilkan nilai pada setiap level LTEX *Effort Multiplier*, rentang dimulai dari 1.2 sampai 0.84. Namun dalam penggunaannya pada *Membership Function* diisi Range[0.84,1.3] karena rentang pada *editor* ini harus berisi nilai yang semakin besar. Kemudian setiap puncak pada LTEX *membership function* diisi sesuai dengan *nilai effort multiplier*. Gambar 4.8. menampilkan implementasi *output* pada LTEX *Membership Function*. Dikarenakan pada input sudah menggunakan deskripsi *Very Low* hingga *High*, pada *output* untuk mempermudah maka *Very Low* ditulis dalam *Decreased Significantly*, *Low* menjadi *Decreased*, dst.

Rating Levels	Very Low	Low	Nominal	High	Very High
Effort Multipliers	1.20	1.09	1.00	0.91	0.84

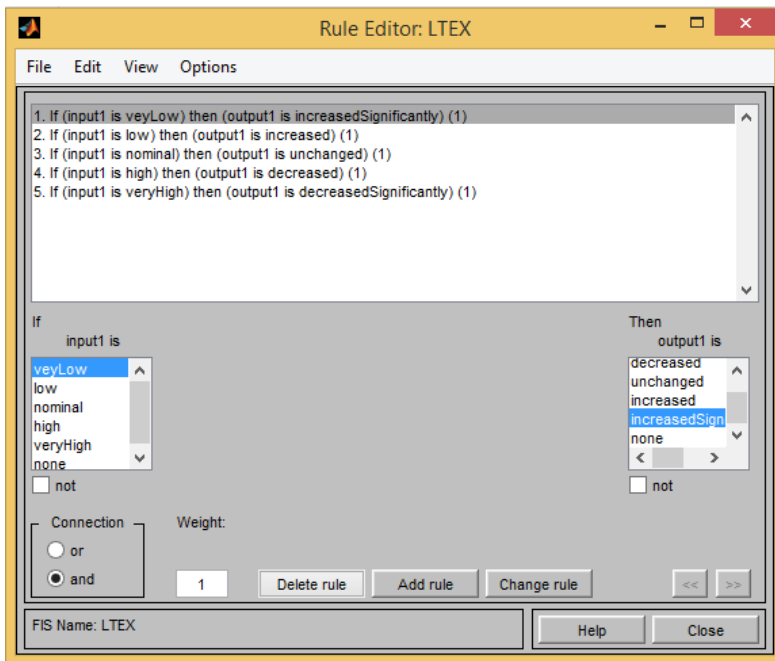
Gambar 4. 7. Nilai LTEX *Effort Multiplier*



Gambar 4. 8. LTEX *Membership Function*

vi. Mengisi *Rule Editor*

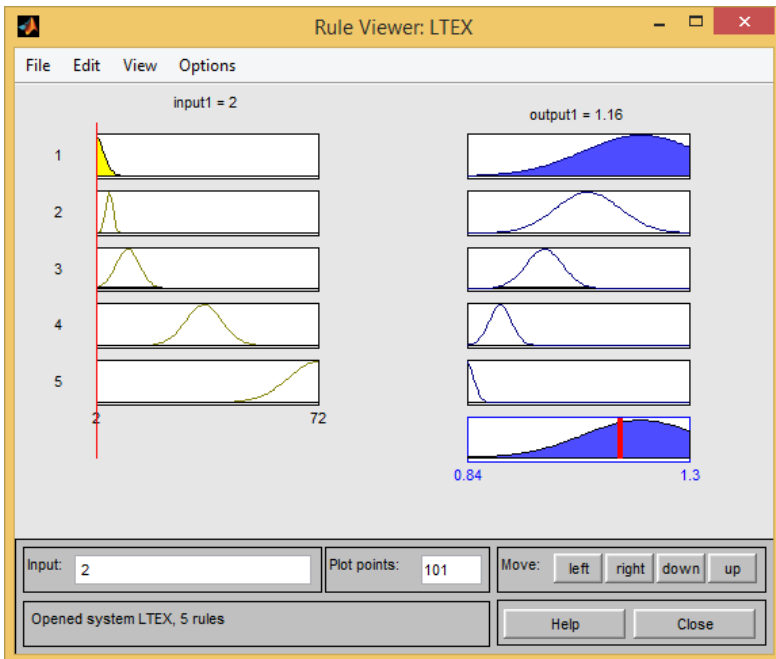
Rule Editor memetakan antara *input* dan *output*. Gambar 4.9. memetakan input yang terdapat pada LTEX dengan *output*. *Very Low* pada input dipetakan dengan *Increased Significantly*. Dikarenakan pada *output* sebelumnya *Very Low* pada *output* dipetakan menjadi *Increased Significantly* maka pada *rule editor* ini, *Very Low* pada *input* dipetakan dengan *Increased Significantly* pada *output*.



Gambar 4. 9. Rule Editor LTEX

vii. Melihat Hasil *Proses Fuzzy Inference*

Setelah *input*, *output* dan *rule editor* diisi, selanjutnya hasil *effort multiplier* bisa ditampilkan pada *rule viewer*. Gambar 4.10 menampilkan hasil proses *fuzzy inference* pada LTEX.



Gambar 4. 10. Rule Viewer LTEX

Tabel 4.1. menunjukkan *effort multiplier* yang standar pada COCOMO II, sedangkan Tabel 4.2. menunjukkan *effort multiplier* yang sudah berubah dengan adanya implementasi *fuzzy logic*. Selanjutnya data *effort multiplier* yang terdapat pada Tabel 4.2. digunakan untuk menghitung estimasi biaya perangkat lunak. Dengan berubahnya nilai *effort multiplier* dengan menggunakan metode *fuzzy logic* pada COCOMO II maka nilai estimasi secara otomatis juga akan berubah mengikuti perubahan nilai *effort multiplier*.

Tabel 4. 1. COCOMO II *Effort Multiplier*

Effort Multiplier	Very Low	Low	Nominal	High	Very High	Extra High
acap	1.42	1.19	1	0.85	0.71	-

Effort Multiplier	Very Low	Low	Nominal	High	Very High	Extra High
pcap	1.34	1.15	1	0.88	0.76	-
aexp	1.22	1.1	1	0.88	0.81	-
modp	1.24	1.1	1	0.91	0.82	-
tool	1.17	1.09	1	0.9	0.78	-
vexp	1.21	1.1	1	0.9	-	-
lexp	1.19	1.09	1	0.91	0.85	-
sced	1.43	1.14	1	1	1	-
stor	-	-	1	1.05	1.17	1.46
data	-	0.9	1	1.14	1.28	-
time	-	-	1	1.11	1.29	1.63
turn	-	0.87	1	1.07	1.15	-
virt	-	0.87	1	1.15	1.3	-
cplx	0.73	0.87	1	1.17	1.34	1.74
rely	0.82	0.92	1	1.1	1.26	-

Tabel 4. 2. *Fuzzy COCOMO II Effort Multiplier*

Effort Multiplier	Very Low	Low	Nominal	High	Very High	Extra High
acap	1.36	1.31	1.13	0.95 4	0.853	-
pcap	1.34	1.14	1	0.88	0.775	-
aexp	1.12	1.1	0.999	0.88 3	0.825	-
modp	1.24	1.1	1	0.91	0.82	-
tool	1.16	1.09	1	0.9	0.78	-
vexp	1.21	1.1	1	0.9	-	-
lexp	1.16	1.1	1	0.90 9	0.85	-
sced	1.4	1.37	1.03	1	1	-

Effort Multiplier	Very Low	Low	Nominal	High	Very High	Extra High
stor	-	-	1	1.07	1.15	1.46
data	-	0.9	1	1.14	1.28	-
time	-	-	1	1.11	1.47	1.6
turn	-	0.87	1	1.07	1.15	-
virt	-	0.87	1	1.14	1.2	-
cplx	0.73	0.87	1	1.17	1.34	1.74
rely	0.82	0.92	1	1.1	1.26	-

4.4. Implementasi *Neural Network* pada *Fuzzy COCOMO II*

Subbab ini menjelaskan bagaimana metode *neural network* diimplementasikan pada *fuzzy COCOMO*. Pada implementasi *neural network* dilakukan dua implementasi arsitektur *neural network*. Hal ini sesuai dengan analisa yang sudah dibahas pada bab sebelumnya. Kode Sumber 4.1. mengimplementasikan *basic neural network* sedangkan Kode Sumber 4.2. mengimplementasikan *modified neural network*.

```

output =0;
HEM = HSF =0;
for(int k=0;k<nVar-2;k++)
{
    HEM += trainVar[i][k]*w[k];
}
HEM += bias1;
HEM = 1 / (1 + (exp(HEM*(-1)))));
for(int k=0;k<5;k++)
{
    HSF += SF[k]*(v[k]+trainVar[i][15]);
}
HSF+=bias2;
HSF = 1 / (1 + (exp(HSF*(-1)))));

output = HEM*wOutput[0]+HSF*wOutput[1]+bias3;

```

```

double err = (trainRes[i]-output);
wOutput[0] = wOutput[0]+lRate*err*HEM;
wOutput[1] = wOutput[1]+lRate*err*HSF;
bias3= bias3 + lRate*err;
double errHEM = err*wOutput[0];
double errHSF = err*wOutput[1];
for(int k=0;k<nVar-2;k++)
{
    w[k] = w[k]+lRate*errHEM*trainVar[i][k];
}
for(int k=0;k<5;k++)
{
    v[k] = v[k]+lRate*errHSF*SF[k];
}
bias1 = bias1+lRate*errHEM;
bias2 = bias2+lRate*errHSF;

```

Kode Sumber 4. 1. Fungsi Feed-forward back-propagation pada basic neural network

```

output =0;
HEM = HSF =0;
for(int k=0;k<nVar-2;k++)
{
    HEM += trainVar[i][k]*w[k];
}
HEM += bias1;
HEM = 1 / (1 + (exp(HEM*(-1)))));
for(int k=0;k<5;k++)
{
    HSF += SF[k]*(v[k]+trainVar[i][15]);
}
HSF+=bias2;
HSF = 1 / (1 + (exp(HSF*(-1)))));

output = HEM*wOutput[0]+HSF*wOutput[1]+bias3;
double err = (trainRes[i]-output);
double deltaBiasOut = lRate*err;
double wOutput2[2];
wOutput2[0] = lRate*err*HEM;
wOutput2[1] = lRate*err*HSF;
double out2[2];

```



```

out2[0] = err*wOutput[0]*HEM*(1-HEM);
out2[1] = err*wOutput[1]*HSF*(1-HSF);
double deltaWHid[15];
double deltaVHid[5];
double deltaBias[2];
for(int k=0;k<nVar-2;k++)
{
    deltaWHid[k]=out2[0]*lRate*trainVar[i][k];
}
deltaBias[0]= out2[0]*lRate;
deltaBias[1]= out2[1]*lRate;
for(int k=0;k<5;k++)
{
    deltaVHid[k]=out2[1]*lRate*SF[k];
}
bias3+=deltaBiasOut;
wOutput[0]+= wOutput2[0];
wOutput[1]+= wOutput2[1];
for(int k=0;k<nVar-2;k++)
{
    w[k]+= deltaWHid[k];
}
bias1+= deltaBias[0];
bias2+= deltaBias[1];
for(int k=0;k<5;k++)
{
    v[k]+= deltaVHid[k];
}

```

Kode Sumber 4. 2. Fungsi Feed-forward back-propagation pada modified neural network

4.5. Implementasi Aplikasi Estimasi Biaya Perangkat Lunak

Subbab ini menjelaskan implementasi perhitungan estimasi biaya perangkat lunak, lama pengerjaan dan jumlah pekerja. Kode Sumber 4.3. mengaplikasi perhitungan pada aplikasi estimasi.

```

double c =
0.91+(0.01*(pmat3+team3+resl3+flex3+prec3));
double effort = 2.94 * acap3 * pcap3 *aexp3 *
modp3 * tool3 * vexp3 * lexp3 * sced3 * stor3 *
data3 * time3 * turn3 * virt3 * cplx3 * rely3 *
Math.Pow(Convert.ToDouble(kloc.Text),c);
double f = 0.28 + 0.2 * 0.01 * (pmat3 + team3 +
resl3 + flex3 + prec3);
double tdev = c * Math.Pow(effort, f);
if (kloc.Text == "" || cost.Text == "")
    MessageBox.Show("Silahkan isi LOC dan gaji
perbulan");
Else
{
    groupBox4.Visible = true;
    jlh.Text = Convert.ToString(effort/152);
    lama.Text = Convert.ToString(tdev);
    double temp = Convert.ToDouble(cost.Text);
    double temp2 = effort/152*tdev*temp;
    biaya.Text = Convert.ToString(temp2);
}

```

Kode Sumber 4. 3. Implementasi fungsi perhitungan estimasi biaya perangkat lunak

BAB V

PENGUJIAN DAN EVALUASI

Bab ini membahas uji coba dan evaluasi pada metode yang dikembangkan.

5.1. Lingkungan Uji Coba

Implementasi dilakukan pada lingkungan sebagai berikut:

a. Perangkat Keras

Uji coba dilakukan pada sebuah laptop dengan spesifikasi:

- i. *Processor* : Intel® Core™ i5-2410M Processor
2.30GHz with Turbo Boost up to 2.90GHz
- ii. *RAM* : 4 GB
- iii. *HDD* : 500 GB

b. Perangkat Lunak

Uji coba dilakukan dengan perangkat lunak sebagai berikut:

- i. *Operating System* : Windows 8.1 64-bit
- ii. Implementasi COCOMO : Microsoft Excel
- iii. Implementasi *Fuzzy* : Matlab R2013a
- iv. Implementasi *Neural Network* : Dev-C++
- v. Implementasi Aplikasi : Visual Studio 2012 64-bit

5.2. Metode Pengujian

Pengujian metode yang dikembangkan dilakukan dengan cara mencari akurasi dari setiap metode. Metode yang paling sering digunakan dalam rangka mencari akurasi adalah *Mean Relative Error* (MRE) dan *Mean Magnitude Relative Error* (MMRE) [22]. MRE adalah metode untuk mencari persentasi akurasi antara data yang sebenarnya dengan data yang diprediksi sedangkan MMRE merupakan rata-rata dari MRE pada n data. Secara umum nilai 0% pada MRE dan MMRE menyatakan bahwa estimasi sama dengan nilai sebenarnya. Persamaan 6.1. merupakan persamaan MRE sedangkan 6.2. merupakan persamaan MMRE.

$$MRE = \frac{|Actual\ Effort - Estimated\ Effort|}{Actual\ Effort} * 100 \quad \dots(6.1.)$$

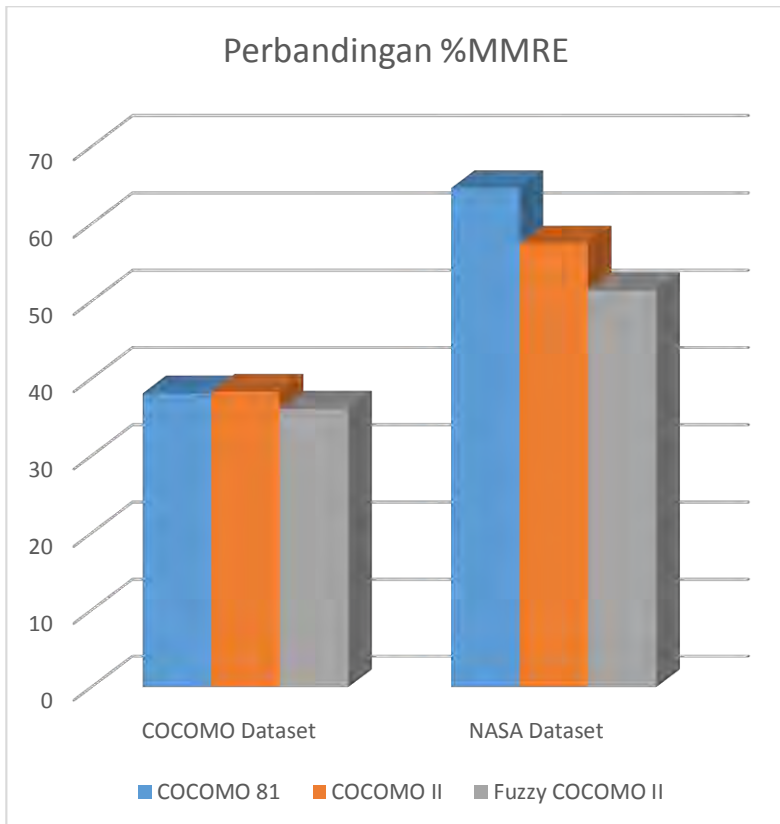
$$MMRE = \frac{1}{N} \sum_{i=1}^N MRE_i \quad \dots(6.2.)$$

5.3. Evaluasi COCOMO 81, COCOMO II, dan Fuzzy COCOMO II

Berdasarkan hasil implementasi pada bab sebelumnya, dengan membandingkan *actual effort* dan *estimated effort* pada setiap *data point* maka didapat MRE. Lampiran D, E, dan F menampilkan hasil perbandingan antar nilai sebenarnya dan nilai estimasi dengan menggunakan COCOMO 81, COCOMO II serta *fuzzy-COCOMO II* terhadap *dataset* COCOMO dan NASA. Tabel 6.1. menampilkan MMRE dari ketiga metode yaitu COCOMO 81, COCOMO II dan fuzzy COCOMO II terhadap dua *dataset*. Tabel 5.1. menampilkan perbandingan MMRE dengan tiga metode yang digunakan. Gambar 5.1. menampilkan perbandingan MMRE secara grafik agar lebih mudah dicermati.

Tabel 5. 1. Perbandingan MMRE terhadap COCOMO 81, COCOMO II dan *fuzzy* COCOMO II

Metode	COCOMO dataset (%MMRE)	NASA dataset (%MMRE)
COCOMO 81	38.08	64.61
COCOMO II	38.29	57.46
Fuzzy COCOMO II	36.11	51.42



Gambar 5. 1. Grafik perbandingan MMRE terhadap COCOMO 81, COCOMO II dan *fuzzy* COCOMO II

5.4. Evaluasi *Neural Network* pada *Fuzzy COCOMO*

Setiap *dataset* dibagi menjadi dua bagian yaitu data *training* dan data *testing*. Data *training* dimasukkan kedalam arsitektur *neural network* untuk selanjutnya dilakukan iterasi dalam rangka mencari nilai dari setiap *weight*. Lalu data *testing* dijalankan menurut arsitektur dan *weight* yang sudah ada. Pada pengujian ini dataset dibagi menjadi 80% data *training* dan 20%

data *testing*. Tabel 5.2. menunjukkan perbandingan antara COCOMO dengan 2 arsitektur *neural network* berdasarkan data *testing*, dimana pemilihan data *testing* dilakukan dengan cara acak pada *dataset* COCOMO. Tabel 5.3. menunjukkan perbandingan antara COCOMO dengan 2 arsitektur berdasarkan data *testing*, dimana pemilihan data *testing* dilakukan dengan cara acak pada *dataset* NASA.

Tabel 5. 2. Perbandingan MRE COCOMO dengan *neural network* pada *dataset* COCOMO

Proj ect ID	Actual Effort (PM)	MRE (%) menggunakan				
		COCOM O 81 Model	COCOM O II Model	Fuzzy COCOM O II Model	Basic Neural Network	Modified Neural Network
2	1600	8.52	50.87	41.59	33.94	35.68
6	43	38.53	51.92	53.48	7.09	0.02
15	61	31.27	40.32	26.42	7.86	1.91
18	11400	44.96	71.25	56.6	45.31	36.74
25	523	59.33	58.46	44.01	20.55	16.28
26	387	35.91	57.67	42.38	24.36	28.01
29	7.3	25.42	18.64	17.38	2.85	40.6
39	8	12.2	8.5	17.47	5.95	21.38
40	8	23.85	0.75	17.59	82.03	47.49
42	45	184.82	73.75	98.3	10.30	2.13
45	106	15.17	26.65	15.71	12.98	18.32
52	41	6.28	17.11	1.75	14.05	1.86
62	38	19.51	6.41	5.82	26.6	25.49
MMRE(%)		38.91	37.1	33.73	22.61	21.22

Tabel 5. 3. Perbandingan MRE COCOMO dengan *neural network* pada *dataset* COCOMO

Project ID	Actual Effort (PM)	MRE (%) menggunakan				
		COCO MO 81 Model	COCOM O II Model	Fuzzy COCOM O II Model	Basic Neural Network	Modified Neural Network
4	36	22.72	44.57	24.89	8.92	6.15
9	72	36.85	61.45	56.15	2.42	21.18
10	72	54.19	69	62.83	4.22	3.77
14	215	121.34	24.13	39.14	24.27	51.32
20	60	134.17	26.14	46.15	5.03	43.58
27	70	11.17	37.94	0.6	2.88	0
30	62	18.47	41.76	5.6	5.9	2.94
37	60	1.69	25.47	13.34	0.66	8.73
39	42	15.66	34.48	23.82	8.82	6.79
43	1368	16.7	57.64	51.29	43.17	32.87
44	973	39.13	31.65	20.71	39.35	14.34
55	370	21.98	58.11	50.15	27.72	13.9
59	4560	442.25	74.18	90.26	48	45.16
63	162	11.11	19.28	6.47	20.64	7.14
64	150	59.03	61.26	55.11	19.42	8.78
70	432	42.47	30.63	7.82	33.72	21.5
73	300	31.88	26.90	2.86	25.77	24.52
79	409	21.75	63.37	58.19	27.93	4.22
90	8211	15.76	81.25	79.30	51.37	30.26
MMRE(%)		58.86	45.74	36.56	21.06	18.27

5.5. Uji Coba pada Aplikasi Estimasi Biaya Perangkat Lunak

Pada subbab ini akan dibahas uji coba terhadap aplikasi untuk setiap kasus pengguna yang ada.

5.5.1. Uji Coba Memasukkan Nilai Atribut

Uji coba yang pertama dilakukan adalah ujicoba memasukkan nilai atribut. Tabel 5.4. menampilkan blackbox testing terhadap kasus pengguna memasukkan nilai atribut. Sedangkan Gambar 5.2. adalah ujicoba aplikasi dalam menangani adanya kesalahan.

Tabel 5. 4. *Blackbox Testing* Kasus Pengguna Memasukkan Nilai Atribut

No .	Skenario Pengujian	Test Case	Hasil yang diharapkan	Hasil Pengujian	Kesimpulan
1.	Menggan ti nilai atribut	Menggan ti nilai atribut Time menjadi High	Aplikasi dapat menggan ti atribut yang diinginkan	Sesuai Harapan	Valid
2.	Menekan tombol <i>calculate</i> tanpa mengisi <i>field</i> KLOC dan Cost	KLOC:- Cost:-	Aplikasi meminta pengguna mengisi semua field	Sesuai Harapan	Valid
3.	Menekan tombol	KLOC: 100	Aplikasi meminta	Sesuai Harapan	Valid

No .	Skenario Pengujian	Test Case	Hasil yang diharapkan	Hasil Pengujian	Kesimpulan
	<i>calculate</i> dengan hanya mengisi LOC dan tidak mengisi <i>cost</i>	Cost: -	pengguna mengisi semua field		
4.	Menekan tombol <i>calculate</i> dengan hanya mengisi <i>Cost</i> dan tidak mengisi LOC	KLOC: - Cost:100	Aplikasi meminta pengguna mengisi semua field	Sesuai Harapan	Valid

Software Cost Estimation

Effort Multipliers

Analyst Capability	Nominal	Programmers Capability	Nominal	Application Experience	Nominal
Modern Programming Practices	Nominal	Use of Software Tools	Nominal	Virtual Machine Experience	Nominal
Language Experience	Nominal	Schedule Constraint	Nominal	Main Memory Constraint	Nominal
Data Base Size	Nominal	Time Constraint for CPU	Nominal	Turnaround time	Nominal
Machine Volatility	Nominal	Process Complexity	Nominal		

Scale Factors

Precededness	Nominal	Process Maturity	Nominal
Development Flexibility	Nominal	Team Cohesion	Nominal

KLOC: Cost (person/month): Rp

Silahkan isi LOC dan gaji perbulan

OK

Gambar 5. 2. Peringatan field masih ada yang belum diisi pada aplikasi

5.5.2. Uji Coba Melihat Nilai Estimasi Biaya Perangkat Lunak

Uji coba yang kedua dilakukan adalah ujicoba melihat hasil estimasi biaya perangkat lunak. Tabel 5.5. menampilkan blackbox testing terhadap kasus pengguna memasukkan nilai atribut. Sedangkan Gambar 5.3. adalah ujicoba aplikasi dalam menangani adanya kesalahan.

Tabel 5. 5. *Blackbox* Testing Kasus Pengguna Memasukkan Nilai Atribut

No	Skenario Pengujian	Test Case	Hasil yang diharapkan	Hasil Pengujian	Kesimpulan
1.	Input sudah dilakukan aplikasi menampilkan hasil estimasi	Seluruh atribut dibuat nominal KLOC : 100 Cost: 5000000	Aplikasi menampilkan estimasi biaya aplikasi	Sesuai Harapan	Valid

Software Cost Estimation

Effort Multipliers

Analyst Capability: Nominal Programmers Capability: Nominal Application Experience: Nominal

Modern Programming Practices: Nominal Use of Software Tools: Nominal Virtual Machine Experience: Nominal

Language Experience: Nominal Schedule Constraint: Nominal Main Memory Constraint: Nominal

Data Base Size: Nominal Time Constraint for CPU: Nominal Turnaround time: Nominal

Machine Volatility: Nominal Process Complexity: Nominal Required Software Reliability: Nominal

Scale Factors

Precetdness: Nominal Process Maturity: Nominal Architecture / Risk Resolution: Nominal

Development Flexibility: Nominal Team Cohesion: Nominal

KLOC: 100 Cost (person/month): 5000000 Rp **CALCULATE**

Result

Employee: 3.56 persons Schedule: 8.13 months Price: 144,754,406.61 Rupiah

Gambar 5. 3. Aplikasi menampilkan jumlah pegawai, lama serta biaya perangkat lunak

Hasil pengujian menunjukkan bahwa mengestimasi biaya perangkat lunak menggunakan COCOMO 81 maupun COCOMO II masih kurang akurat, masih perlu ditambahkan adanya metode yang meningkatkan akurasi tersebut. *Fuzzy logic* dan *artificial neural network* pada pengujian terbukti mampu meningkatkan akurasi akurasi pada COCOMO II. Peningkatan yang didapat dengan menggunakan kedua metode ini adalah 43% untuk *dataset* COCOMO dan 60% untuk *dataset* NASA.

BAB VI

KESIMPULAN DAN SARAN

Pada bab ini akan diberikan kesimpulan yang diambil selama pengerjaan tugas akhir serta saran-saran tentang pengembangan yang dapat dilakukan terhadap tugas akhir ini di masa yang akan datang.

6.1 Kesimpulan

Dari hasil selama proses perancangan, implementasi, serta pengujian dapat diambil kesimpulan sebagai berikut:

1. Estimasi biaya perangkat lunak berhasil dimodelkan dengan cara memasukkan atribut sesuai kebutuhan perangkat lunak yaitu *effort multiplier* dan jumlah baris kode sebuah perangkat lunak.
2. Metode *fuzzy* dapat digunakan untuk meningkatkan akurasi pada COCOMO dengan peningkatan akurasi sebesar 5% pada COCOMO *dataset* dan 20.41 % pada NASA *dataset*.
3. Metode *artificial neural network* dapat diimplementasikan bersama dengan metode *fuzzy logic* pada COCOMO II. Penggabungan kedua metode ini merupakan metode yang paling baik dalam estimasi biaya perangkat lunak dengan tingkat akurasi sebesar 81,79% pada COCOMO *dataset* dan 78.78% pada NASA *dataset*.

6.2 Saran

Berikut saran-saran untuk pengembangan dan perbaikan sistem di masa yang akan datang. Di antaranya adalah sebagai berikut:

1. Optimasi estimasi biaya perangkat lunak pada COCOMO dengan *fuzzy logic* dapat ditingkatkan dengan cara menambahkan *scale factor* dan LOC sebagai *input* pada *fuzzy set*.

2. Optimasi estimasi biaya perangkat lunak pada COCOMO dengan *neural network* dapat ditingkatkan dengan cara menemukan arsitektur baru yang lebih sesuai serta parameter-parameter yang sesuai pula.

DAFTAR PUSTAKA

- [1] C. S. Reddy dan K. Raju, "An improved fuzzy approach for COCOMO's effort estimation using gaussian membership function," *Journal of Software*, vol. 4, pp. 452-459, 2009.
- [2] R. S. Pressman, *Software Engineering: a practitioner's approach*, New York: McGraw-Hill, 1997.
- [3] P. Bourque dan R. Dupuis, *Guide to the Software Engineering Body of Knowledge (SWEBOK)*, IEEE Computer Society, 2004.
- [4] M. O. Saliu, *Adaptive Fuzzy Logic based Framework for Software Development Effort Prediction*, Dhahran, 2003.
- [5] J. T. Marchewka, *Information Technology Project Management*, John Wiley & Sons, Inc., 2009.
- [6] C. Kirsopp, M. Shepperd dan J. Hart, "Search heuristics, case-based reasoning and software project effort prediction," dalam *Genetic and Evolutionary Computation Conference*, New York, 2002.
- [7] S. Chulani, B. Boehm dan B. Steece, *Calibrating Software Cost Models Using Bayesian Analysis*, University of Southern California Center for Software Engineering, 1998.
- [8] L. H. Putnam, "A General Empirical Solution to the Macro Software Sizing and Estimating Problem," *IEEE Transactions on Software Engineering*, vol. 4, no. 4, pp. 345-361, 1978.
- [9] K. Srinivasan dan D. Fisher, "Machine Learning Approaches to Estimating Software

Development Effort,” *IEEE Transactions on Software Engineering*, vol. 21, no. 2, 1995.

- [10] L. A. Zadeh, “Fuzzy Logic, Neural Networks, and Soft Computing,” *Communication of the ACM*, vol. 37, no. 3, pp. 77-84, 1994.
- [11] B. Boehm, *Software Engineering Economics*, New Jersey, USA: Prentice-Hall, Englewood Cliffs, 1994.
- [12] B. Boehm, *Software Cost Estimation with COCOMO II*, New Jersey, USA: Prentice Hall, 2000.
- [13] R. Sarno, C. Djeni, I. Mukhlash dan D. Sunaryono, “Developing a workflow management system for enterprise resource planning,” *Journal of Theoretical and Applied Information Technology*, vol. 72, no. 3, pp. 412-421, 2015.
- [14] L. Zadeh, “Fuzzy Sets,” *Elsevier*, vol. 8, pp. 338-353, 1965.
- [15] J. R. Jang, “Neuro-Fuzzy and Soft Computing, A Computational Approach to Learning and Machine Intelligence,” *Prentice-Hall, Inc.*, p. 72, 1997.
- [16] M. Negnevitsky, *Artificial Intelligence - A Guide to Intelligent System*, England: Addison-Wesley, 2002.
- [17] V. Chandra, “Software effort estimation: a fuzzy logic approach,” *International Journal of Computer Applications*, vol. 103, pp. 39-43, 2014.
- [18] M. Jorgenson dan D. Sjoberg, “The impact of customer expectation on software development effort estimates,” *International Journal of Project Management*, vol. 22, no. 4, pp. 317-325.
- [19] B. Boehm, B. Clark, E. Horowitz, R. Madachy, R. Shelby dan C. Westland, “Cost models

- for future software life cycle processes: COCOMO 2.0,” *Annals of Software Engineering*, 1995.
- [20] I. Attarzadeh dan S. Ow, “Improving the accuracy of software cost estimation model on a new fuzzy logic model,” *IDOSI*, pp. 177-184, 2010.
- [21] J. Starczewski, “Defuzzification of Uncertain Fuzzy Sets,” dalam *Studies in Fuzziness and Soft Computing*, New York, Springer, 2013, pp. 77-135.
- [22] L. Briand, K. Emam, D. Surmann dan I. Wiecek, “An assessment and comparison of common software cost estimation modeling techniques,” *ISERN*, 1998.
- [23] A. Kaushik, A. Soni dan R. Soni, “A simple neural network approach to software cost estimation,” *Global Journal of Computer Science and Technology*, vol. 13, no. 1, pp. 23 - 30, 2013.

LAMPIRAN

A. COCOMO Dataset

N o.	REL Y	DAT A	CPL X	TI ME	STO R	VIR T	TUR N	AC AP	AE XP	PC AP	VE XP	LE XP	MO DP	TO OL	SCE D	LO C	ACTU AL
1	0.88	1.16	0.7	1	1.06	1.15	1.07	1.19	1.13	1.17	1.1	1	1.24	1.1	1.04	113	2040
2	0.88	1.16	0.85	1	1.06	1	1.07	1	0.91	1	0.9	0.95	1.1	1	1	293	1600
3	1	1.16	0.85	1	1	0.87	1	0.86	0.82	0.86	0.9	0.95	0.91	0.91	1	132	243
4	0.75	1.16	0.7	1	1	0.87	1	1.19	0.91	1.42	1	0.95	1.24	1	1.04	60	240
5	0.88	0.94	1	1	1	0.87	1	1	1	0.86	0.9	0.95	1.24	1	1	16	33
6	0.75	1	0.85	1	1.21	1	1	1.46	1	1.42	0.9	0.95	1.24	1.1	1	4	43
7	0.75	1	1	1	1	0.87	0.87	1	1	1	0.9	0.95	0.91	0.91	1	6.9	8
8	1.15	0.94	1.3	1.66	1.56	1.3	1	0.71	0.91	1	1.21	1.14	1.1	1.1	1.08	22	1075
9	1.15	0.94	1.3	1.3	1.21	1.15	1	0.86	1	0.86	1.1	1.07	0.91	1	1	30	423
10	1.4	0.94	1.3	1.11	1.56	1	1.07	0.86	0.82	0.86	0.9	1	1	1	1	29	321
11	1.4	0.94	1.3	1.11	1.56	1	1.07	0.86	0.82	0.86	0.9	1	1	1	1	32	218
12	1.15	0.94	1.3	1.11	1.06	1	1	0.86	0.82	0.86	1	0.95	0.91	1	1.08	37	201
13	1.15	0.94	1.3	1.11	1.06	1.15	1	0.71	1	0.7	1.1	1	0.82	1	1	25	79

N o.	REL Y	DAT A	CPL X	TI ME	STO R	VIR T	TUR N	AC AP	AE XP	PC AP	VE XP	LE XP	MO DP	TO OL	SCE D	LO C	ACTU AL
14	1.15	0.94	1.65	1.3	1.56	1.15	1	0.86	1	0.7	1.1	1.07	1.1	1.24	1.23	3	60
15	1.4	0.94	1.3	1.3	1.06	1.15	0.87	0.86	1.13	0.86	1.21	1.14	0.91	1	1.23	3.9	61
16	1.4	1	1.3	1.3	1.56	1	0.87	0.86	1	0.86	1	1	1	1	1	6.1	40
17	1.4	1	1.3	1.3	1.56	1	0.87	0.86	0.82	0.86	1	1	1	1	1	3.6	9
18	1.15	1.16	1.15	1.3	1.21	1	1.07	0.86	1	1	1	1	1.24	1.1	1.08	320	11400
19	1.15	1.08	1	1.11	1.21	0.87	1	0.71	0.91	1	1	1	0.91	0.91	1	115 0	6600
20	1.4	1.08	1.3	1.11	1.21	1.15	1.07	0.71	0.82	1.17	1.1	1.07	1.24	1	1.08	299	6400
21	1	1.16	1.15	1.11	1.21	0.87	0.87	0.86	1	1	1	1	0.91	0.91	1	252	2455
22	1.15	1	1	1.3	1.06	1	1	0.86	0.82	0.86	0.9	1	0.91	1	1.23	118	724
23	1.15	1	1	1.11	1.06	1	1	0.86	0.82	0.86	0.9	1	1	1	1.23	77	539
24	0.88	1	0.85	1.11	1.06	1	0.87	1	1.29	1	1.1	0.95	0.82	0.83	1	90	453
25	1.15	1.16	1.3	1.15	1.06	1	0.87	0.86	1	0.86	1.1	1	0.82	0.91	1.08	38	523
26	1	1	0.85	1.11	1.06	1.15	1.07	0.86	1	0.86	1.1	1	0.91	1.1	1.08	48	387
27	1.15	0.94	1.15	1.3	1.21	1	0.87	1	1	1	1	1	0.82	1.1	1.08	9.4	88
28	1.15	1.08	1.3	1.11	1.21	1.15	1.07	0.86	1	0.86	1.1	1.07	1.1	1.1	1	13	98
29	0.88	1	1	1	1	1	1.19	1.1	1.29	0.86	1	1	0.91	0.91	1.23	2.1 4	7.3

N o.	REL Y	DAT A	CPL X	TI ME	STO R	VIR T	TUR N	AC AP	AE XP	PC AP	VE XP	LE XP	MO DP	TO OL	SCE D	LO C	ACTU AL
30	0.88	1	1	1	1	1	1	1	1.29	0.86	1	1	0.91	0.91	1.23	1.9 8	5.9
31	1.4	1.08	1	1.11	1.56	1.15	1.07	0.86	0.82	0.86	1.1	1.07	1	1	1	62	1063
32	0.88	1.08	0.85	1	1	1	1	0.71	0.82	1	1	1	1.1	1.1	1	390	702
33	1.4	1.08	1.3	1.6	1.56	1.15	1	0.86	0.82	0.86	0.9	1	0.91	0.91	1	42	605
34	1.15	1.08	1	1.11	1	1	0.87	1	1	1	1	1	0.91	1.1	1.23	23	230
35	0.75	0.94	1.3	1.11	1.21	1.15	1	1	0.91	1	1.1	1	1.24	1.24	1	13	82
36	0.88	1.08	0.81	1	1	0.87	0.87	1.19	1	1.17	0.9	0.95	1	0.91	1.04	15	55
37	0.88	0.94	0.7	1	1.06	1	1	0.86	0.82	0.86	1	1	1	1	1	60	47
38	1	1	1.15	1	1	0.87	0.87	0.71	0.91	1	0.9	0.95	0.82	0.91	1	15	12
39	1	1	1.15	1	1	0.87	1	0.71	0.82	0.7	1	0.95	0.91	1.1	1	6.2	8
40	1	0.94	1.3	1	1	1	0.87	0.86	0.82	1.17	1	1	1.1	1	1	3	8
41	0.88	0.94	1	1	1	0.87	0.87	1	0.82	0.7	0.9	0.95	0.91	0.91	1	5.3	6
42	0.88	1.08	1	1	1.06	0.87	1.07	0.86	1	1	0.9	0.95	0.91	0.91	1.04	45. 5	45
43	1	1.08	1	1	1.21	0.87	1.07	0.86	1	1	0.9	0.95	1	1	1.04	28. 6	83
44	0.88	1.08	1	1.11	1.21	0.87	1.07	1	1	1	0.9	0.95	1.1	1	1.04	30. 6	87
45	0.88	1.08	1	1	1.06	0.87	1.07	1	1	1	0.9	0.95	1	0.91	1.04	35	106

N o.	REL Y	DAT A	CPL X	TI ME	STO R	VIR T	TUR N	AC AP	AE XP	PC AP	VE XP	LE XP	MO DP	TO OL	SCE D	LO C	ACTU AL
46	0.88	1.08	1	1	1.06	0.87	1.07	1	1	0.86	0.9	0.95	1	1	1.04	73	126
47	0.75	0.94	1.3	1	1	0.87	0.87	0.71	0.82	0.7	1.1	1.07	1.1	1	1.04	23	36
48	0.88	0.94	0.85	1	1	0.87	1	1.19	0.91	1.17	0.9	0.95	1.1	1	1.04	464	1272
49	1	1	0.85	1	1	1	0.87	0.71	1	0.7	1.1	1	0.82	0.91	1	91	156
50	1.15	1	1	1.3	1.21	1	0.87	0.86	1	0.86	1.1	1	1	1	1	24	176
51	0.88	1	1	1	1	1	1.15	1.19	1	1.42	1	0.95	1.24	1.1	1.04	10	122
52	0.88	0.94	0.85	1	1.06	1.15	1	1	1	1	1.1	1.07	1.24	1.1	1	8.2	41
53	0.88	0.94	1.15	1.11	1.21	1.3	1	0.71	1	0.7	1.1	1.07	1	1.1	1.08	5.3	14
54	1	0.94	1	1	1.06	1.15	0.87	1	0.82	1	1	0.95	0.91	1.1	1	4.4	20
55	0.88	0.94	0.7	1	1	0.87	0.87	0.86	0.82	1.17	0.9	0.95	1.1	1	1	6.3	18
56	1.15	0.94	1.3	1.3	1.21	1	1	0.86	0.91	1	1.1	1.07	1.1	1.1	1.08	27	958
57	1	0.94	1.15	1.11	1.21	1.3	1	1	1	1	1.1	1.07	1.1	1.1	1.23	17	237
58	1.4	0.94	1.3	1.66	1.21	1	1	0.71	0.82	0.7	0.9	0.95	0.91	1	1	25	130
59	1	0.94	1.15	1.11	1.06	1	0.87	1	1	1	1	1	0.91	1	1	23	70
60	1.15	0.94	1.3	1.11	1.06	1	1	0.86	1.13	0.86	1.1	1.07	1.1	1.1	1.08	6.7	57
61	1	0.94	1.15	1	1	0.87	0.87	0.86	1	0.86	0.9	1	0.82	1	1	28	50
62	0.88	0.94	1.3	1.11	1.21	1.15	1	0.86	0.82	0.7	1.21	1.14	0.91	1.24	1	9.1	38

N o.	REL Y	DAT A	CPL X	TI ME	STO R	VIR T	TUR N	AC AP	AE XP	PC AP	VE XP	LE XP	MO DP	TO OL	SCE D	LO C	ACTU AL
63	1	0.94	1.15	1	1	1	0.87	0.71	0.82	0.86	1	1	0.82	1	1	10	15

B. NASA Dataset

N o.	N a m e	Categ ory	f / g	Ce nte r	Y ea r	Mode	R E L Y	D A T A	C P L X	TI ME	ST O R	VI R T	T U R N	A C A P	A E X P	P C A P	V E X P	L E X P	M O DP	T O O L	S C E D	L O C	AC TU AL
1	de	avionics monitoring	g	2	1979	semidetached	H	l	h	n	n	l	l	n	n	n	n	h	h	n	l	25.9	117.6
2	de	avionics monitoring	g	2	1979	semidetached	H	l	h	n	n	l	l	n	n	n	n	h	h	n	l	24.6	117.6
3	de	avionics monitoring	g	2	1979	semidetached	H	l	h	n	n	l	l	n	n	n	n	h	h	n	l	7.7	31.2
4	de	avionics monitoring	g	2	1979	semidetached	H	l	h	n	n	l	l	n	n	n	n	h	h	n	l	8.2	36
5	de	avionics monitoring	g	2	1979	semidetached	H	l	h	n	n	l	l	n	n	n	n	h	h	n	l	9.7	25.2

N o .	N a m e	Catog y	f / g	Ce nte r	Y ea r	Mode	R E L Y	D A T A	C P L X	T I M E	S T O R	V I R T	T U R N	A C A P	A E X P	P C A P	V E X P	L E X P	M O DP	T O O L	S C E D	L O C	AC TU AL
6	de	avionics monitori ng	g	2	1 9 7 9	semid etach ed	H	l	h	n	n	l	l	n	n	n	n	h	h	n	l	2. 2	8.4
7	de	avionics monitori ng	g	2	1 9 7 9	semid etach ed	H	l	h	n	n	l	l	n	n	n	n	h	h	n	l	3. 5	10.8
8	er b	avionics monitori ng	g	2	1 9 8 2	semid etach ed	H	l	h	n	n	l	l	n	n	n	n	h	h	n	l	66 .6	352. 8
9	ga l	missionp lanning	g	1	1 9 8 0	semid etach ed	H	l	h	xh	xh	l	h	h	h	h	n	h	h	h	n	7. 5	72
1 0	ga l	missionp lanning	g	1	1 9 8 0	semid etach ed	N	l	h	n	n	l	l	h	vh	vh	n	h	n	n	n	20	72
1 1	ga l	missionp lanning	g	1	1 9 8 4	semid etach ed	N	l	h	n	n	l	l	h	vh	h	n	h	n	n	n	6	24
1 2	ga l	missionp lanning	g	1	1 9 8 0	semid etach ed	N	l	h	n	n	l	l	h	vh	vh	n	h	n	n	n	10 0	360
1 3	ga l	missionp lanning	g	1	1 9	semid etach ed	N	l	h	n	n	l	l	h	vh	n	n	l	n	n	n	11 .3	36

N o .	N a m e	Cat eg o r y	f / g	Ce n t e r	Y e a r	Mode	R E L Y	D A T A	C P L X	T I M E	S T O R	V I R T	T U R N	A C A P	A E X P	P C A P	V E X P	L E X P	M O D P	T O O L	S C E D	L O C	AC T U AL
					85																		
14	gal	missionplanning	g	1	1980	semidetached	N	l	h	n	n	h	l	h	h	h	l	vl	n	n	n	100	215
15	gal	missionplanning	g	1	1983	semidetached	N	l	h	n	n	l	l	h	vh	h	n	h	n	n	n	20	48
16	gal	missionplanning	g	1	1982	semidetached	N	l	h	n	n	l	l	h	n	n	n	vl	n	n	n	100	360
17	gal	missionplanning	g	1	1980	semidetached	N	l	h	n	xh	l	l	h	vh	vh	n	h	n	n	n	150	324
18	gal	missionplanning	g	1	1984	semidetached	N	l	h	n	n	l	l	h	h	h	n	h	n	n	n	31.5	60
19	gal	missionplanning	g	1	1983	semidetached	N	l	h	n	n	l	l	h	vh	h	n	h	n	n	n	15	48
20	gal	missionplanning	g	1	1984	semidetached	N	l	h	n	xh	l	l	h	h	n	n	h	n	n	n	32.5	60

N o .	N a m e	Cat eg o r y	f / g	C e n t e r	Y e a r	Mode	R E L Y	D A T A	C P L X	T I M E	S T O R	V I R T	T U R N	A C A P	A E X P	P C A P	V E X P	L E X P	M O D P	T O O L	S C E D	L O C	AC T U AL
21	X	avionics monit ing	g	2	1985	semid etach ed	H	l	h	n	n	l	l	n	n	n	n	h	h	n	l	19.7	60
22	X	avionics monit ing	g	2	1985	semid etach ed	H	l	h	n	n	l	l	n	n	n	n	h	h	n	l	66.6	300
23	X	simulat ion	g	2	1985	semid etach ed	H	l	h	n	n	l	l	n	n	n	n	h	h	n	l	29.5	120
24	X	monitor_ control	g	2	1986	semid etach ed	H	n	n	h	n	n	n	n	h	h	n	n	n	n	n	15	90
25	X	monitor_ control	g	2	1986	semid etach ed	H	n	h	n	n	n	n	n	h	h	n	n	n	n	n	38	210
26	X	monitor_ control	g	2	1986	semid etach ed	N	n	n	n	n	n	n	n	h	h	n	n	n	n	n	10	48
27	X	realdata p rocess ing	g	2	1982	semid etach ed	N	vh	h	vh	vh	l	h	vh	h	n	l	h	vh	vh	l	15.4	70
28	X	realdata p rocess ing	g	2	19	semid etach ed	N	vh	h	vh	vh	l	h	vh	h	n	l	h	vh	vh	l	48.5	239

N o .	N a m e	Cat eg o r y	f / g	C e n t e r	Y e a r	M o d e	R E L Y	D A T A	C P L X	T I M E	S T O R	V I R T	T U R N	A C A P	A E X P	P C A P	V E X P	L E X P	M O D P	T O O L	S C E D	L O C	A C T U A L
					8 2																		
2 9	X	realdata rocessin g	g	2	1 9 8 2	semid etach ed	N	vh	h	vh	vh	l	h	vh	h	n	l	h	vh	vh	l	16 .3	82
3 0	X	commun ications	g	2	1 9 8 2	semid etach ed	N	vh	h	vh	vh	l	h	vh	h	n	l	h	vh	vh	l	12 .8	62
3 1	X	batchdat aprocess ing	g	2	1 9 8 2	semid etach ed	N	vh	h	vh	vh	l	h	vh	h	n	l	h	vh	vh	l	32 .6	170
3 2	X	datacapt ure	g	2	1 9 8 2	semid etach ed	N	vh	h	vh	vh	l	h	vh	h	n	l	h	vh	vh	l	35 .5	192
3 3	X	missionp lanning	g	2	1 9 8 5	semid etach ed	H	l	h	n	n	l	l	n	n	n	n	h	h	n	l	5. 5	18
3 4	X	avionics monitori ng	g	2	1 9 8 7	semid etach ed	H	l	h	n	n	l	l	n	n	n	n	h	h	n	l	10 .4	50
3 5	X	avionics monitori ng	g	2	1 9 8 7	semid etach ed	H	l	h	n	n	l	l	n	n	n	n	h	h	n	l	14	60

N o .	N a m e	Cat eg o r y	f / g	C e n t e r	Y e a r	Mode	R E L Y	D A T A	C P L X	T I M E	S T O R	V I R T	T U R N	A C A P	A E X P	P C A P	V E X P	L E X P	M O D P	T O O L	S C E D	L O C	AC T U AL
36	X	monitor_ control	g	2	1986	semid etach ed	H	n	h	n	n	n	n	n	n	n	n	n	n	n	n	6.5	42
37	X	monitor_ control	g	2	1986	semid etach ed	N	n	h	n	n	n	n	n	n	n	n	n	n	n	n	13	60
38	X	monitor_ control	g	2	1986	semid etach ed	N	n	h	n	n	n	n	n	n	h	n	h	h	h	n	90	444
39	X	monitor_ control	g	2	1986	semid etach ed	N	n	h	n	n	n	n	n	n	n	n	n	n	n	n	8	42
40	X	monitor_ control	g	2	1986	semid etach ed	N	n	h	h	n	n	n	n	n	n	n	n	n	n	n	16	114
41	hs t	datacapt ure	g	2	1980	semid etach ed	N	h	h	vh	h	l	h	h	n	h	l	h	h	n	l	177.9	1248
42	sl p	launchpr ocessing	g	6	1975	semid etach ed	H	l	h	n	n	l	l	n	n	h	n	n	h	vl	n	302	2400
43	Y	applicati on_grou nd	g	5	19	semid etach ed	N	h	l	n	n	h	n	h	h	n	n	n	h	h	n	282.1	1368

N o .	N a m e	Cat eg o r y	f / g	C e n t e r	Y e a r	Mode	R E L Y	D A T A	C P L X	T I M E	S T O R	V I R T	T U R N	A C A P	A E X P	P C A P	V E X P	L E X P	M O D P	T O O L	S C E D	L O C	AC TU AL
					82																		
44	Y	applicati on_grou nd	g	5	1982	semid etach ed	H	h	l	n	n	n	h	h	h	n	n	n	h	n	n	284.7	973
45	Y	avionics monitori ng	g	5	1982	semid etach ed	H	h	n	n	n	l	l	n	h	h	n	h	n	n	n	79	400
46	Y	avionics monitori ng	g	5	1977	semid etach ed	L	n	n	n	n	l	l	h	h	vh	n	h	l	l	h	423	2400
47	Y	missionp lanning	g	5	1977	semid etach ed	N	n	n	n	n	l	n	h	vh	vh	l	h	h	n	n	190	420
48	Y	missionp lanning	g	5	1984	semid etach ed	N	n	h	n	h	n	n	h	h	n	n	h	h	n	h	47.5	252
49	Y	missionp lanning	g	5	1980	semid etach ed	Vh	n	xh	h	h	l	l	n	h	n	n	n	l	h	n	21	107
50	Y	simulati on	g	5	1983	semid etach ed	N	h	h	vh	n	n	h	h	h	h	n	h	l	l	h	78	571.4

N o .	N a m e	Cat eg o r y	f / g	C e n t e r	Y e a r	Mode	R E L Y	D A T A	C P L X	T I M E	S T O R	V I R T	T U R N	A C A P	A E X P	P C A P	V E X P	L E X P	M O D P	T O O L	S C E D	L O C	AC T U A L
51	Y	simulati on	g	5	1 9 8 4	semid etach ed	N	h	h	vh	n	n	h	h	h	h	n	h	l	l	h	11 .4	98.8
52	Y	simulati on	g	5	1 9 8 5	semid etach ed	N	h	h	vh	n	n	h	h	h	h	n	h	l	l	h	19 .3	155
53	Y	missionp lanning	g	5	1 9 7 9	semid etach ed	H	n	vh	h	h	l	h	h	n	n	h	h	l	vh	h	10 1	750
54	Y	missionp lanning	g	5	1 9 7 9	semid etach ed	H	n	h	h	h	l	h	n	h	n	n	n	l	vh	n	21 9	212 0
55	Y	utility	g	5	1 9 7 9	semid etach ed	H	n	h	h	h	l	h	n	h	n	n	n	l	vh	n	50	370
56	sp l	datacapt ure	g	2	1 9 7 9	semid etach ed	Vh	h	h	vh	vh	n	n	vh	vh	vh	n	h	h	h	l	22 7	118 1
57	sp l	batchdat aprocess ing	g	2	1 9 7 7	semid etach ed	N	h	vh	n	n	l	n	h	n	vh	l	n	h	n	l	70	278
58	de	avionics monitori ng	g	2	1 9	semid etach ed	H	l	h	n	n	l	l	n	n	n	n	h	h	n	l	0. 9	8.4

N o .	N a m e	Cat eg o r y	f / g	C e n t e r	Y e a r	Mode	R E L Y	D A T A	C P L X	T I M E	S T O R	V I R T	T U R N	A C A P	A E X P	P C A P	V E X P	L E X P	M O D P	T O O L	S C E D	L O C	AC T U AL
					7 9																		
5 9	sl p	operatin gsystem	g	6	1 9 7 4	semid etach ed	Vh	l	xh	xh	vh	l	l	h	vh	h	vl	h	vl	vl	h	98 0	456 0
6 0	sl p	operatin gsystem	g	6	1 9 7 5	embe dded	N	l	h	n	n	l	l	vh	n	vh	h	h	n	l	n	35 0	720
6 1	Y	operatin gsystem	g	5	1 9 7 6	embe dded	H	n	xh	h	h	l	l	h	n	n	h	h	h	h	n	70	458
6 2	Y	utility	g	5	1 9 7 9	embe dded	H	n	xh	h	h	l	l	h	n	n	h	h	h	h	n	27 1	246 0
6 3	Y	avionics monitori ng	g	5	1 9 7 1	organ ic	N	n	n	n	n	l	l	h	h	h	n	h	n	l	n	90	162
6 4	Y	avionics monitori ng	g	5	1 9 8 0	organ ic	N	n	n	n	n	l	l	h	h	h	n	h	n	l	n	40	150
6 5	Y	avionics monitori ng	g	5	1 9 7 9	embe dded	H	n	h	h	n	l	l	h	h	h	n	h	n	n	n	13 7	636

N o .	N a m e	Cat eg o r y	f / g	Ce n t e r	Y ea r	Mode	R E L Y	D A T A	C P L X	T I M E	S T O R	V I R T	T U R N	A C A P	A E X P	P C A P	V E X P	L E X P	M O D P	T O O L	S C E D	L O C	AC TU AL
66	Y	avionics monitori ng	g	5	1977	embed ded	H	n	h	h	n	h	l	h	h	h	n	h	n	vl	n	150	882
67	Y	avionics monitori ng	g	5	1976	embed ded	Vh	n	h	h	n	l	l	h	h	h	n	h	n	n	n	339	444
68	Y	avionics monitori ng	g	5	1983	organ ic	L	h	l	n	n	h	l	h	h	h	n	h	n	l	n	240	192
69	Y	avionics monitori ng	g	5	1978	semid etach ed	H	n	h	n	vh	l	n	h	h	h	h	h	l	l	l	144	576
70	Y	avionics monitori ng	g	5	1979	semid etach ed	N	l	n	n	vh	l	n	h	h	h	h	h	l	l	l	151	432
71	Y	avionics monitori ng	g	5	1979	semid etach ed	N	l	h	n	vh	l	n	h	h	h	h	h	l	l	l	34	72
72	Y	avionics monitori ng	g	5	1979	semid etach ed	N	n	h	n	vh	l	n	h	h	h	h	h	l	l	l	98	300
73	Y	avionics monitori ng	g	5	19	semid etach ed	N	n	h	n	vh	l	n	h	h	h	h	h	l	l	l	85	300

N o .	N a m e	Cat eg o r y	f / g	C e n t e r	Y e a r	M o d e	R E L Y	D A T A	C P L X	T I M E	S T O R	V I R T	T U R N	A C A P	A E X P	P C A P	V E X P	L E X P	M O D P	T O O L	S C E D	L O C	A C T U A L
					7 9																		
7 4	Y	avionics monit oring	g	5	1 9 8 2	semid etach ed	N	l	n	n	vh	l	n	h	h	h	h	h	l	l	l	20	240
7 5	Y	avionics monit oring	g	5	1 9 7 8	semid etach ed	N	l	n	n	vh	l	n	h	h	h	h	h	l	l	l	11 1	600
7 6	Y	avionics monit oring	g	5	1 9 7 8	semid etach ed	H	vh	h	n	vh	l	n	h	h	h	h	h	l	l	l	16 2	756
7 7	Y	avionics monit oring	g	5	1 9 7 8	semid etach ed	H	h	vh	n	vh	l	n	h	h	h	h	h	l	l	l	35 2	120 0
7 8	Y	operatin gsystem	g	5	1 9 7 9	semid etach ed	H	n	vh	n	vh	l	n	h	h	h	h	h	l	l	l	16 5	97
7 9	Y	missionp lanning	g	5	1 9 8 4	embe dded	H	n	vh	h	h	l	vh	h	n	n	h	h	h	vh	h	60	409
8 0	Y	missionp lanning	g	5	1 9 8 4	embe dded	H	n	vh	h	h	l	vh	h	n	n	h	h	h	vh	h	10 0	703

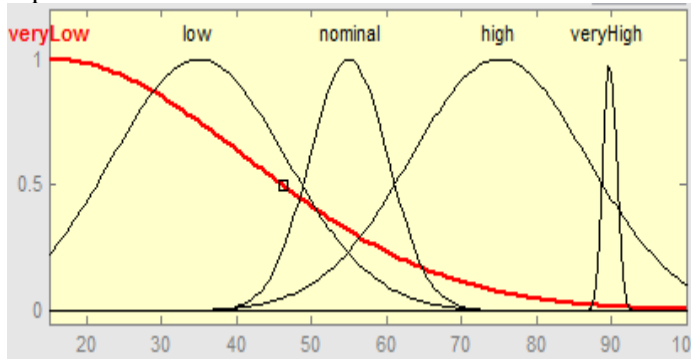
N o .	N a m e	Cat eg o r y	f / g	Ce n t e r	Y e a r	Mode	R E L Y	D A T A	C P L X	T I M E	S T O R	V I R T	T U R N	A C A P	A E X P	P C A P	V E X P	L E X P	M O D P	T O O L	S C E D	L O C	AC T U A L
81	hs t	Avionics	f	2	1980	em be dded	H	vh	vh	xh	xh	h	h	n	n	n	l	l	n	n	h	32	1350
82	hs t	Avionics	f	2	1980	em be dded	H	h	h	vh	xh	h	h	h	h	h	h	h	h	n	n	53	480
83	sp l	Avionics	f	3	1977	em be dded	H	l	vh	vh	xh	l	n	vh	vh	vh	vl	vl	h	h	n	41	599
84	sp l	Avionics	f	3	1977	em be dded	H	l	vh	vh	xh	l	n	vh	vh	vh	vl	vl	h	h	n	24	430
85	Y	Avionics	f	5	1977	em be dded	Vh	h	vh	xh	xh	n	n	h	h	h	h	h	h	n	h	165	4178.2
86	Y	science	f	5	1977	em be dded	Vh	h	vh	xh	xh	n	n	h	h	h	h	h	h	n	h	65	1772.5
87	Y	Avionics	f	5	1977	em be dded	Vh	h	vh	xh	xh	n	l	h	h	h	h	h	h	n	h	70	1645.9
88	Y	Avionics	f	5	19	em be dded	Vh	h	xh	xh	xh	n	n	h	h	h	h	h	h	n	h	50	1924.5

N o .	N a m e	Cat eg o r y	f / g	Ce n t e r	Y e a r	Mode	R E L Y	D A T A	C P L X	T I M E	S T O R	V I R T	T U R N	A C A P	A E X P	P C A P	V E X P	L E X P	M O D P	T O O L	S C E D	L O C	AC T U AL
					7 7																		
8 9	ga l	Avionics	f	5	1 9 8 2	embe dded	Vh	l	vh	vh	xh	l	l	h	l	n	vl	l	l	h	h	7. 25	648
9 0	Y	Avionics	f	5	1 9 8 0	embe dded	Vh	h	vh	xh	xh	n	n	h	h	h	h	h	h	n	h	23 3	821 1
9 1	X	Avionics	f	2	1 9 8 3	embe dded	H	n	vh	vh	vh	h	h	n	n	n	l	l	n	n	h	16 .3	480
9 2	X	Avionics	f	2	1 9 8 3	embe dded	H	n	vh	vh	vh	h	h	n	n	n	l	l	n	n	h	6. 2	12
9 3	X	science	f	2	1 9 8 3	embe dded	H	n	vh	vh	vh	h	h	n	n	n	l	l	n	n	h	3	38

C. Fuzzy Inference System (FIS)

1. ACAP

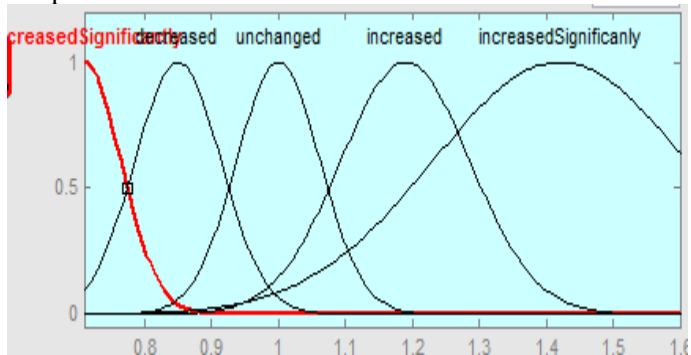
Input:



Rules:

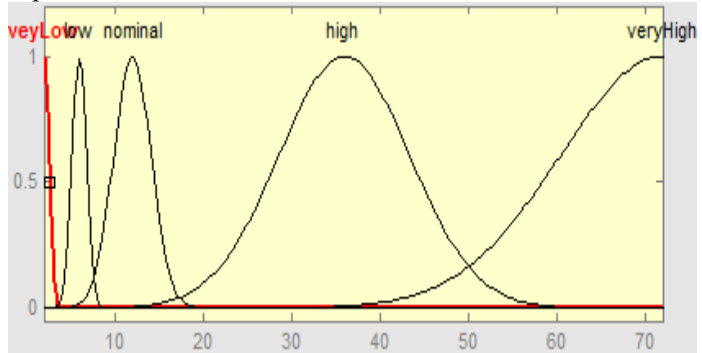
1. If (input1 is veryLow) then (output1 is increasedSignificantly) (1)
2. If (input1 is low) then (output1 is increased) (1)
3. If (input1 is nominal) then (output1 is unchanged) (1)
4. If (input1 is high) then (output1 is decreased) (1)
5. If (input1 is veryHigh) then (output1 is decreasedSignificantly) (1)

Output:



2. APEX

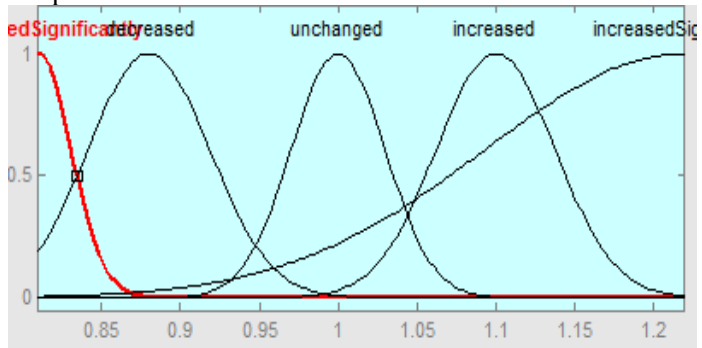
Input:



Rules:

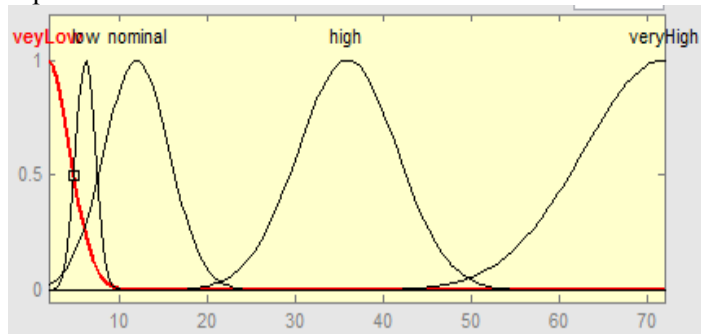
1. If (input1 is veyLow) then (output1 is increasedSignificantly) (1)
2. If (input1 is low) then (output1 is increased) (1)
3. If (input1 is nominal) then (output1 is unchanged) (1)
4. If (input1 is high) then (output1 is decreased) (1)
5. If (input1 is veryHigh) then (output1 is decreasedSignificantly) (1)

Output:



3. LTEX

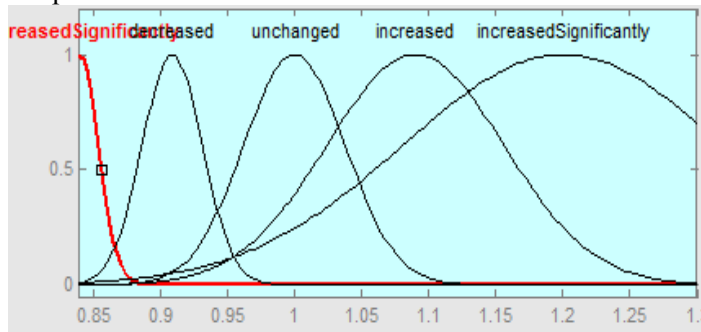
Input:



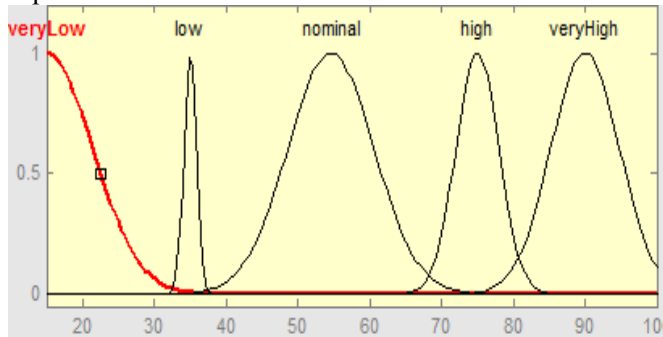
Rules:

1. If (input1 is veryLow) then (output1 is increasedSignificantly) (1)
2. If (input1 is low) then (output1 is increased) (1)
3. If (input1 is nominal) then (output1 is unchanged) (1)
4. If (input1 is high) then (output1 is decreased) (1)
5. If (input1 is veryHigh) then (output1 is decreasedSignificantly) (1)

Output:



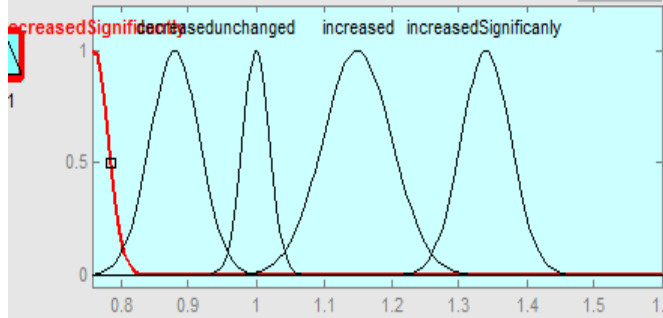
Input:



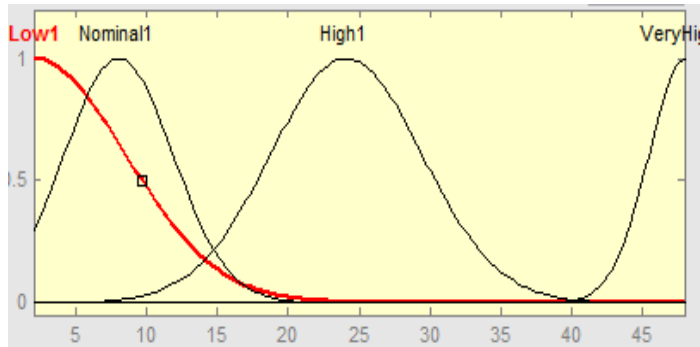
Rules:

1. If (input1 is veryLow) then (output1 is increasedSignificantly) (1)
2. If (input1 is low) then (output1 is increased) (1)
3. If (input1 is nominal) then (output1 is unchanged) (1)
4. If (input1 is high) then (output1 is decreased) (1)
5. If (input1 is veryHigh) then (output1 is decreasedSignificantly) (1)

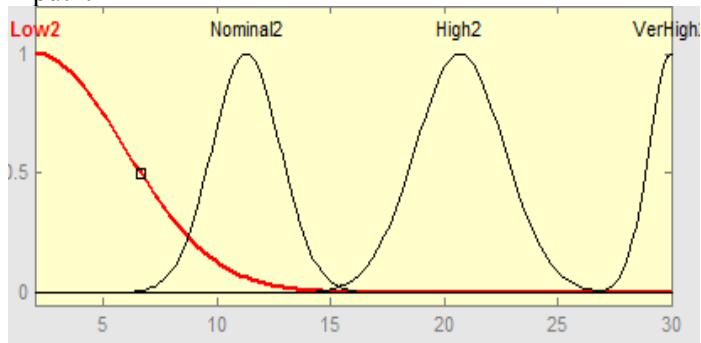
Output:



5. PVOL
Input 1:



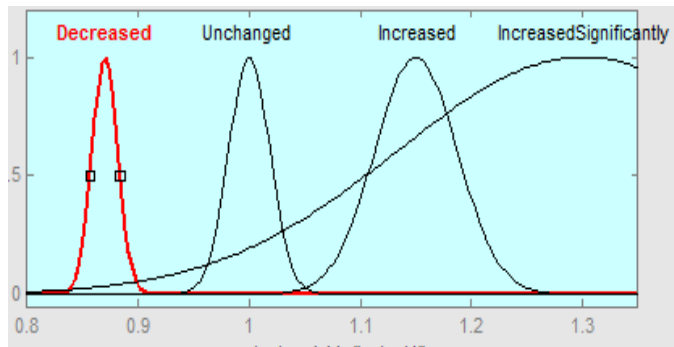
Input2:



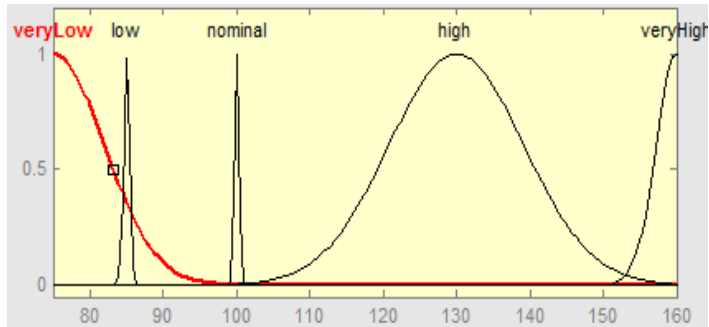
Rules:

1. If (major is VeryHigh1) and (minor is VerHigh2) then (output1 is Decreased) (1)
2. If (major is High1) and (minor is High2) then (output1 is Unchanged) (1)
3. If (major is Nominal1) and (minor is Nominal2) then (output1 is Increased) (1)
4. If (major is Low1) and (minor is Low2) then (output1 is IncreasedSignificantly) (1)

Output:



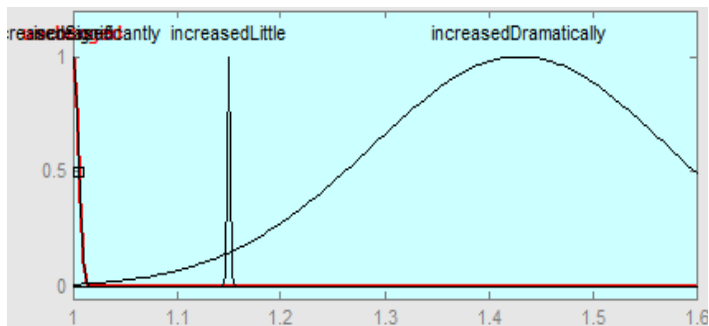
6. SCED
Input:



Rules:

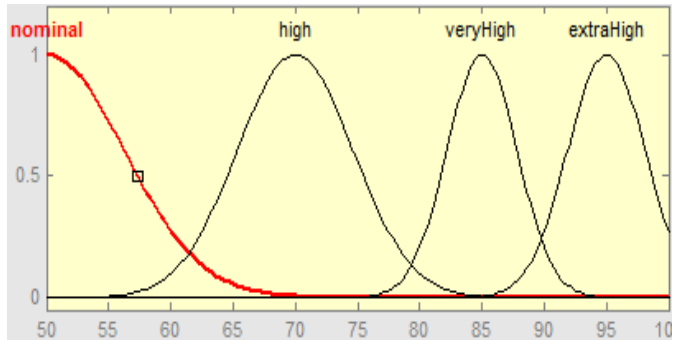
1. If (input1 is veryLow) then (output1 is increasedDramatically) (1)
2. If (input1 is low) then (output1 is increasedLittle) (1)
3. If (input1 is nominal) then (output1 is unchanged) (1)
4. If (input1 is high) then (output1 is increased) (1)
5. If (input1 is veryHigh) then (output1 is increasedSignificantly) (1)

Output:



7. STOR

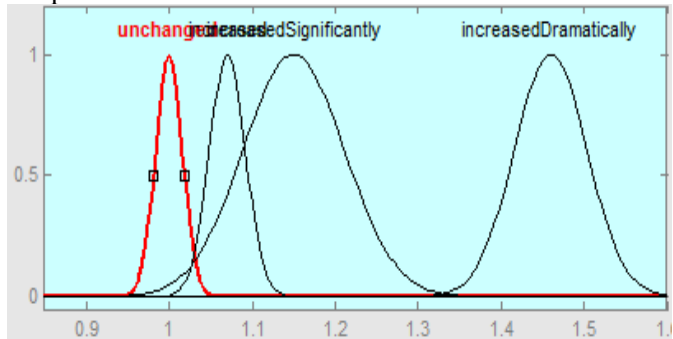
Input:



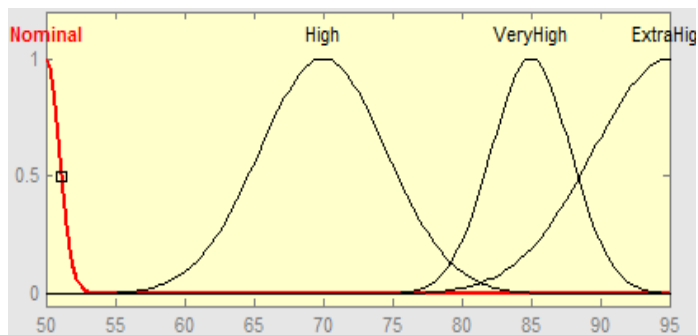
Rules:

1. If (input1 is nominal) then (output1 is unchanged) (1)
2. If (input1 is high) then (output1 is increased) (1)
3. If (input1 is veryHigh) then (output1 is increasedSignificantly) (1)
4. If (input1 is extraHigh) then (output1 is increasedDramatically) (1)

Output:



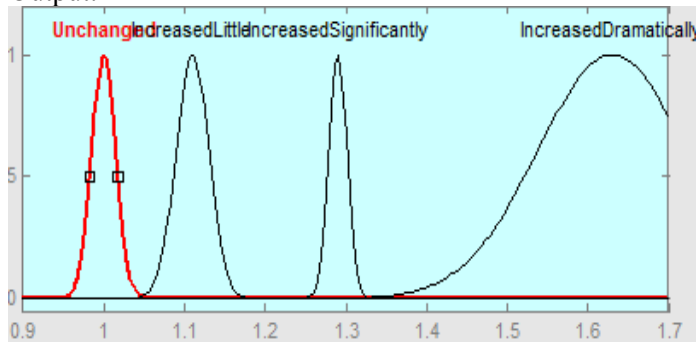
8. TIME
Input:



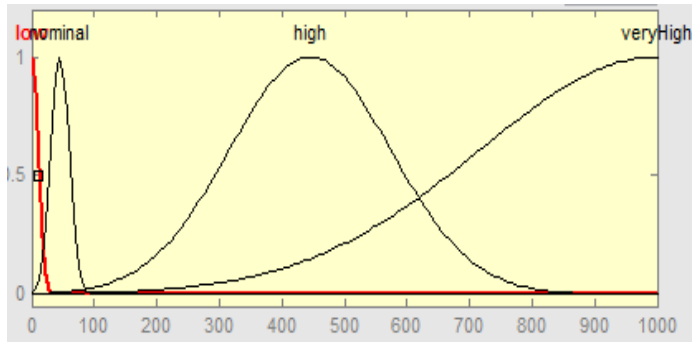
Rules:

1. If (input1 is Nominal) then (output1 is Unchanged) (1)
2. If (input1 is High) then (output1 is IncreasedLittle) (1)
3. If (input1 is VeryHigh) then (output1 is IncreasedSignificantly) (1)
4. If (input1 is ExtraHigh) then (output1 is IncreasedDramatically) (1)

Output:



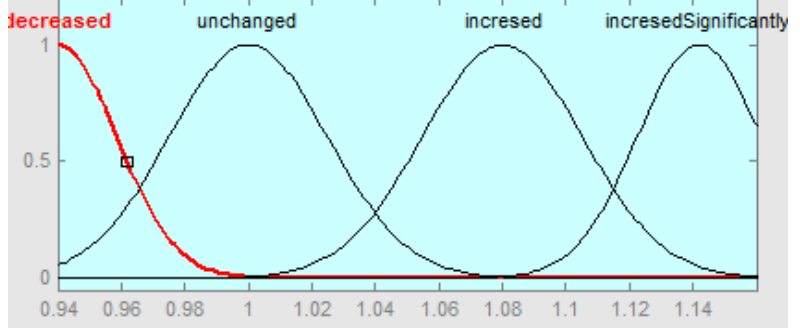
9. DATA
Input:



Rules:

1. If (data is low) then (effort is decreased) (1)
2. If (data is nominal) then (effort is unchanged) (1)
3. If (data is high) then (effort is increased) (1)
4. If (data is veryHigh) then (effort is increasedSignificantly) (1)

Output:



D. Hasil estimasi biaya perangkat lunak menggunakan metode COCOMO 81.

1. Menggunakan dataset COCOMO

Nomor	Actual Effort	Estimated Effort
1	2040	1367.877

Nomor	Actual Effort	Estimated Effort
2	1600	1463.735
3	243	262.0716
4	240	293.6585
5	33	43.92904
6	43	26.43113
7	8	10.49013
8	1075	526.9614
9	423	272.5884
10	321	225.4711
11	218	251.7521
12	201	160.2893
13	79	94.07467
14	60	50.77675
15	61	41.92675
16	40	53.99408
17	9	24.52728
18	11400	6274.686
19	6600	6240.718
20	6400	6200.662
21	2455	1417.666
22	724	607.5415
23	539	353.4076
24	453	325.4772
25	523	212.701
26	387	248.0209
27	88	61.15419
28	98	149.1253
29	7.3	9.155521
30	5.9	6.411284

Nomor	Actual Effort	Estimated Effort
31	1063	701.8885
32	702	1362.375
33	605	503.1924
34	230	148.4429
35	82	115.599
36	55	40.88341
37	47	109.5156
38	12	22.34606
39	8	8.977284
40	8	9.907826
41	6	4.942197
42	45	128.1711
43	83	119.3607
44	87	160.8567
45	106	122.0769
46	126	262.8186
47	36	38.26486
48	1272	2204.634
49	156	141.511
50	176	134.9874
51	122	91.13987
52	41	43.57293
53	14	22.41917
54	20	12.25775
55	18	8.016535
56	958	320.0898
57	237	236.9018
58	130	120.2498
59	70	101.2212

Nomor	Actual Effort	Estimated Effort
60	57	53.67646
61	50	55.95717
62	38	45.41207
63	15	15.27041

2. Menggunakan dataset NASA

Nomor	Actual Effort	Estimated Effort
1	117.6	100.8713
2	117.6	95.21807
3	31.2	25.92637
4	36	27.81913
5	25.2	33.57812
6	8.4	6.373609
7	10.8	10.72083
8	352.8	290.5114
9	72	45.46674
10	72	32.9817
11	24	10.52081
12	360	200.0412
13	36	27.99815
14	215	475.8827
15	48	40.52037
16	360	418.2046
17	324	491.4351
18	60	74.792
19	48	29.35905
20	60	140.5021
21	60	74.2462
22	300	290.5114
23	120	116.7005

Nomor	Actual Effort	Estimated Effort
24	90	62.21629
25	210	182.5637
26	48	30.95003
27	70	62.18187
28	239	224.7353
29	82	66.26599
30	62	50.54937
31	170	144.0271
32	192	158.4515
33	18	17.786
34	50	36.30357
35	60	50.64487
36	42	32.28345
37	60	61.01486
38	444	360.4753
39	42	35.42255
40	114	85.45873
41	1248	1202.737
42	2400	1641.544
43	1368	1139.556
44	973	1353.757
45	400	279.8254
46	2400	1143.731
47	420	436.9307
48	252	194.2495
49	107	170.1225
50	571.4	548.6515
51	98.8	63.66255
52	155	114.8087

Nomor	Actual Effort	Estimated Effort
53	750	602.7024
54	2120	1509.588
55	370	288.6738
56	1181	1236.656
57	278	277.9532
58	8.4	2.342198
59	4560	24726.5
60	720	1555.119
61	458	606.4802
62	2460	3077.954
63	162	143.9988
64	150	61.45643
65	636	937.3165
66	882	1712.897
67	444	3384.476
68	192	430.6556
69	576	821.1095
70	432	615.4906
71	72	133.2667
72	300	463.992
73	300	395.6275
74	240	63.96183
75	600	436.0427
76	756	1086.801
77	1200	2727.923
78	97	1081.092
79	409	497.9532
80	703	919.1937
81	1350	1558.496

Nomor	Actual Effort	Estimated Effort
82	480	787.8587
83	599	358.0491
84	430	188.3022
85	4178.2	4571.831
86	1772.5	1494.877
87	1645.9	1421.498
88	1924.5	1384.886
89	648	133.4219
90	8211	6917.299
91	480	363.2361
92	12	113.8764
93	38	47.6551

E. Hasil estimasi biaya perangkat lunak menggunakan metode COCOMO II.

1. Menggunakan *dataset* COCOMO

Nomor	Actual Effort	Estimated Effort
1	2040	825.8396
2	1600	786.0756
3	243	152.8003
4	240	186.1924
5	33	30.28344
6	43	20.67529
7	8	8.445329
8	1075	326.4309
9	423	165.7464
10	321	122.5045
11	218	135.1773
12	201	96.20011

Nomor	Actual Effort	Estimated Effort
13	79	63.60861
14	60	46.90938
15	61	36.40239
16	40	37.11213
17	9	17.74082
18	11400	3278.062
19	6600	2451.296
20	6400	3030.931
21	2455	759.2659
22	724	366.7816
23	539	226.3123
24	453	167.7378
25	523	217.2651
26	387	163.8207
27	88	42.82884
28	98	110.363
29	7.3	8.660806
30	5.9	6.73384
31	1063	379.2955
32	702	721.4171
33	605	290.488
34	230	116.1499
35	82	79.27595
36	55	61.44552
37	47	67.83092
38	12	14.74838
39	8	7.319988
40	8	8.059899
41	6	4.032352

Nomor	Actual Effort	Estimated Effort
42	45	78.18626
43	83	72.67917
44	87	102.7661
45	106	77.75473
46	126	158.5703
47	36	29.1772
48	1272	927.5519
49	156	88.70502
50	176	83.8572
51	122	61.00625
52	41	33.98328
53	14	20.48868
54	20	8.942176
55	18	6.044466
56	958	194.837
57	237	182.6633
58	130	69.38425
59	70	65.70168
60	57	41.06879
61	50	36.21848
62	38	35.56376
63	15	11.17725

2. Menggunakan *dataset* NASA

Nomor	Actual Effort	Estimated Effort
1	117.6	63.02231
2	117.6	59.85902
3	31.2	18.73636
4	36	19.95301
5	25.2	23.60295

Nomor	Actual Effort	Estimated Effort
6	8.4	5.353246
7	10.8	8.516528
8	352.8	162.0574
9	72	27.75765
10	72	22.31532
11	24	7.751638
12	360	111.5766
13	36	19.87115
14	215	266.8804
15	48	25.83879
16	360	237.0173
17	324	244.3528
18	60	44.21304
19	48	19.37909
20	60	75.68214
21	60	47.93589
22	300	162.0574
23	120	71.78216
24	90	41.69842
25	210	111.3461
26	48	22.76736
27	70	43.44411
28	239	136.8207
29	82	45.98305
30	62	36.10939
31	170	91.9661
32	192	100.1471
33	18	13.38312
34	50	25.30625

Nomor	Actual Effort	Estimated Effort
35	60	34.06611
36	42	24.59457
37	60	44.7174
38	444	203.041
39	42	27.5184
40	114	61.09085
41	1248	683.2473
42	2400	729.3295
43	1368	579.5059
44	973	665.0868
45	400	155.3521
46	2400	537.1428
47	420	231.6398
48	252	106.2675
49	107	104.0292
50	571.4	303.2152
51	98.8	44.31607
52	155	75.02633
53	750	283.6427
54	2120	678.821
55	370	154.982
56	1181	628.6329
57	278	201.6256
58	8.4	2.189964
59	4560	7942.708
60	720	395.0614
61	458	198.1184
62	2460	767.0012
63	162	130.7617

Nomor	Actual Effort	Estimated Effort
64	150	58.1163
65	636	260.8757
66	882	441.742
67	444	739.419
68	192	420.5714
69	576	408.6822
70	432	299.6844
71	72	78.94996
72	300	252.8463
73	300	219.3055
74	240	39.6933
75	600	220.2978
76	756	588.5023
77	1200	1304.337
78	97	536.3226
79	409	149.8181
80	703	249.6968
81	1350	623.2267
82	480	259.9204
83	599	135.0536
84	430	79.05577
85	4178.2	1090.088
86	1772.5	429.4285
87	1645.9	402.3415
88	1924.5	428.9355
89	648	56.36934
90	8211	1539.336
91	480	157.2928
92	12	59.82916

Nomor	Actual Effort	Estimated Effort
93	38	28.94959

F. Hasil estimasi biaya perangkat lunak menggunakan metode *Fuzzy-COCOMO II*.

1. Menggunakan *dataset* COCOMO

Nomor	Actual Effort	Estimated Effort
1	2040	910.3922
2	1600	934.4907
3	243	179.7141
4	240	205.4409
5	33	35.17295
6	43	20.00503
7	8	9.808898
8	1075	417.6927
9	423	214.4811
10	321	144.2406
11	218	159.162
12	201	134.5261
13	79	81.00259
14	60	59.34822
15	61	44.884
16	40	48.84001
17	9	23.80328
18	11400	4947.29
19	6600	2991.67
20	6400	4344.538
21	2455	861.8626
22	724	476.6725
23	539	258.103

Nomor	Actual Effort	Estimated Effort
24	453	182.4407
25	523	292.8381
26	387	222.9877
27	88	65.0782
28	98	125.3259
29	7.3	8.569054
30	5.9	6.838981
31	1063	446.7725
32	702	909.2499
33	605	332.8152
34	230	128.3674
35	82	89.43886
36	55	66.91285
37	47	81.38751
38	12	18.29248
39	8	9.397619
40	8	9.407457
41	6	4.869156
42	45	89.23647
43	83	80.00909
44	87	113.9012
45	106	89.34859
46	126	182.2144
47	36	36.74156
48	1272	1014.542
49	156	111.8226
50	176	108.4706
51	122	67.01726
52	41	40.28229

Nomor	Actual Effort	Estimated Effort
53	14	27.59244
54	20	10.69675
55	18	7.047305
56	958	298.0578
57	237	184.8431
58	130	85.94343
59	70	77.84883
60	57	56.96626
61	50	41.82754
62	38	40.20995
63	15	14.08743

2. Menggunakan dataset NASA

Nomor	Actual Effort	Estimated Effort
1	117.6	85.40365
2	117.6	81.11698
3	31.2	25.39027
4	36	27.03899
5	25.2	31.98515
6	8.4	7.254364
7	10.8	11.54103
8	352.8	219.6094
9	72	31.57056
10	72	26.7639
11	24	9.116992
12	360	133.8195
13	36	23.61157
14	215	299.1434
15	48	30.38997
16	360	266.823
17	324	293.0647

Nomor	Actual Effort	Estimated Effort
18	60	51.22921
19	48	22.79248
20	60	87.69213
21	60	64.95953
22	300	219.6094
23	120	97.27443
24	90	48.69824
25	210	130.0375
26	48	26.58927
27	70	70.41698
28	239	221.7678
29	82	74.53226
30	62	58.5284
31	170	149.0645
32	192	162.3249
33	18	18.13591
34	50	34.29336
35	60	46.16413
36	42	28.59699
37	60	51.99454
38	444	235.8236
39	42	31.99664
40	114	71.03253
41	1248	1067.906
42	2400	840.7697
43	1368	666.361
44	973	771.4771
45	400	181.2314
46	2400	616.1804

Nomor	Actual Effort	Estimated Effort
47	420	277.8174
48	252	121.8218
49	107	123.8064
50	571.4	388.6947
51	98.8	56.80922
52	155	96.17701
53	750	323.7304
54	2120	807.8736
55	370	184.446
56	1181	1054.685
57	278	277.0413
58	8.4	2.967694
59	4560	8675.729
60	720	497.4719
61	458	232.9024
62	2460	901.665
63	162	151.5122
64	150	67.33877
65	636	302.274
66	882	503.0545
67	444	856.7572
68	192	483.0743
69	576	543.0544
70	432	398.2188
71	72	104.9082
72	300	335.9806
73	300	291.4118
74	240	52.74421
75	600	292.7304

Nomor	Actual Effort	Estimated Effort
76	756	781.9984
77	1200	1733.195
78	97	712.6622
79	409	170.9921
80	703	284.9869
81	1350	690.8687
82	480	340.2062
83	599	192.8101
84	430	112.8645
85	4178.2	1203.715
86	1772.5	474.1909
87	1645.9	444.2804
88	1924.5	473.6465
89	648	72.75556
90	8211	1699.792
91	480	198.9599
92	12	75.67799
93	38	36.61838

G. Daftar Istilah

No.	Istilah	Deskripsi
1.	<i>Input</i>	Masukan berupa nilai atau data
2.	<i>Output</i>	Keluaran berupa nilai atau data
3.	<i>Field</i>	Kolom tempat mengisi nilai atau atribut
4.	<i>Testing</i>	Data yang digunakan untuk proses uji coba
5.	<i>Training</i>	Data yang digunakan untuk proses pembelajaran
6.	<i>Dataset</i>	Kumpulan data
7.	<i>Weight</i>	Nilai yang digunakan sebagai pengali antara beberapa atribut
8.	<i>Data point</i>	Setiap bari data yang terdapat pada <i>dataset</i> .
9.	<i>Actual Effort</i>	Nilai usaha yang sebenarnya

10.	<i>Estimated Effort</i>	Nilai estimasi usaha
11.	<i>Membership Function</i>	Fungsi keanggotaan terhadap suatu kurva
12.	<i>FIS</i>	Sebuah alat yang terdapat pada tool Matlab untuk menggambarkan <i>fuzzy set</i>
13.	<i>Fuzzy set</i>	Sebuah kurva yang menggambarkan
14.	<i>Layer</i>	Lapisan
15.	<i>Crisp Data</i>	Data yang bersifat mutlak, biasanya nilai yang selalu sama
16.	<i>Sigmoid</i>	Salah satu model perhitungan aktivasi pada <i>neural network</i>
17.	<i>Node</i>	Setiap inputan pada <i>neural network</i>

(Halaman ini sengaja dikosongkan)

BIODATA PENULIS



Penulis, Johannes Christian Pandapotan Sidabutar lahir di Jakarta pada 06 Agustus 1993. Penulis adalah anak pertama dari dua bersaudara dan dibesarkan di Pematangsiantar, Sumatera Utara.

Penulis menempuh pendidikan formal di TK Slamet Riyadi Jakarta (1997-1999), SD Kalam Kudus Pematangsiantar (1999-2005), SMP Kalam Kudus Pematangsiantar (2005-2008), dan SMA Kalam Kudus Pematangsiantar (2008-2011). Pada tahun 2011, penulis memulai pendidikan S1 jurusan Teknik Informatika Fakultas Teknologi Informasi di Institut Teknologi Sepuluh Nopember, Surabaya, Jawa Timur.

Di jurusan Teknik Informatika, penulis mengambil bidang minat Manajemen Informasi (MI) dan memiliki ketertarikan di bidang *software enterprise*, *business process*, SAP, kecerdasan komputasional. Penulis juga aktif dalam organisasi kemahasiswaan yaitu Himpunan Mahasiswa Teknik *Computer* Informatika (HMTIC) yaitu sebagai staff departemen riset dan teknologi HMTIC 2012/2013. Penulis sempat menjadi asisten dosen selama menjalani perkuliahan di Teknik Informatika yaitu asisten dosen mata kuliah aljabar linier. Penulis dapat dihubungi melalui alamat email jo.chris93@gmail.com.